ATARI ST-LISTING EXTRA

ATARI ST LISTING Extra

DM9,80/ÖS80/SFR9,80

Sonderheft 02/88- The best of Atari Special

Rund 100 Seiten Software für Ihren Atari ST!

Tools
Spiele
Anwender-Programme
Utilities
Lern-Software

Tips & Tricks

ST-Listing

REPORT

LISTINGS

Messebericht Büro & Computer '88

Report

Der Münchner Messe-Flop

Nur ein Prozent aller deutschen Betriebsstätten nutzen derzeit Möglichkeiten der EDV. Die Notwendigkeit des Computereinsatzes haben viele Firmen längst erkannt, einen ersten Schritt aber – bedingt durch fehlende Sachkenntnis und mangelnde Markt-Transparenz – noch nicht getan. Eines der Hauptziele der Münchner Messe Büro & Computer war es, diese Schwellenängste abzubauen.

Bereits zum 14. Mal fand zwischen dem 4. und 7. Mai in München die Messe Büro & Computer statt. Der Besucher, der große internationale Fachmessen wie die CeBIT in Hannover oder die Systems in München gewohnt ist, war sicherlich von dem sich bietenden Bild überrascht.

Der größte Teil der 163 Firmen, die sich auf einer Fläche von 38.000 Quadratmetern präsentierten, waren keine multinationalen Konzerne, sondern regionale Fachhändler. So können auf der Messe geknüpfte Kontakte ohne Probleme weiter vertieft werden. Dies kommt sicherlich dem Ziel dieser Veranstaltung entgegen, Hemmschwellen auf Seiten der EDV-Anwender abzubauen.

Eigens zu diesem Zweck wurde ein in die Messe integriertes Forum geschaffen, das dem Besucher Fachvorträge anbot und als Diskussionsplattform gedacht war.

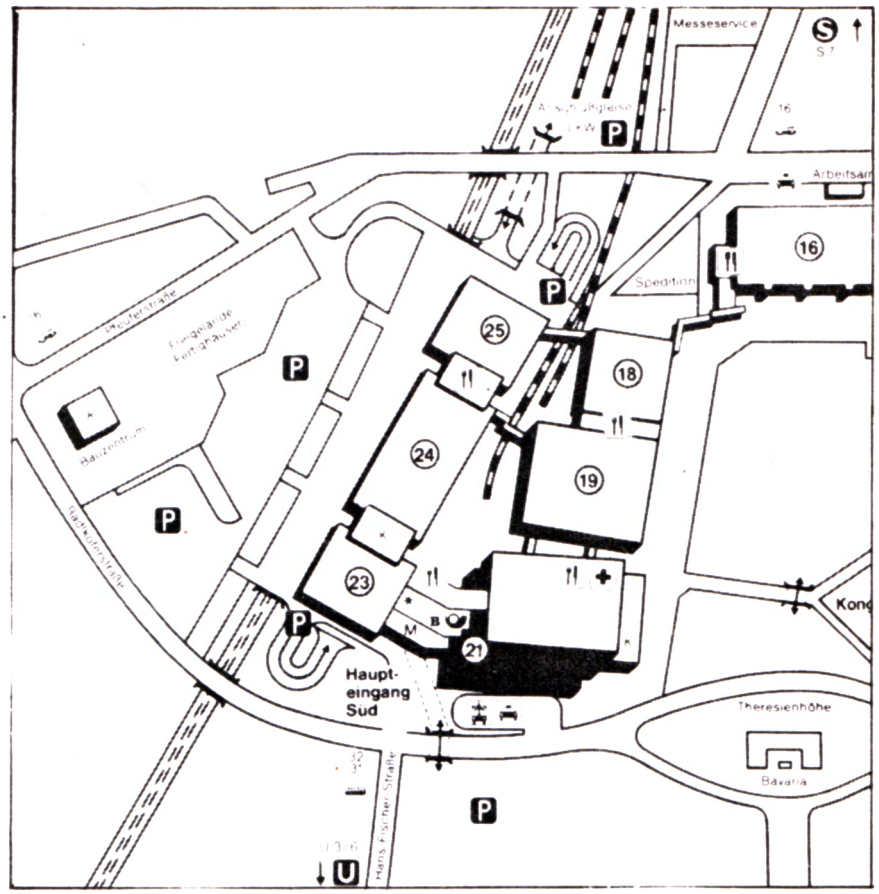
Wir besuchten die Büro & Computer am letzten der insgesamt vier Messetage. Das Getümmel großer Massen gewohnt, überraschte uns zu Beginn die Ruhe in den Hallen. Neben einer sehr großzügigen Standplanung war hierfür auch der schwache Besuch an diesem Tag verantwortlich. Einen „Langen Samstag" opfern wohl nur wenige Leute für einen Messebesuch.

So „drängte" sich an den Ständen der Aussteller nur das Standpersonal. Ein durchaus erfreulicher Aspekt für den Rat und Hilfe suchenden Interessenten, der andernorts mit ein paar Prospekten abgewimmelt wird, sich hier aber in Ruhe informieren konnte.

Neben einer großen Anzahl von Produkten aus dem Bereich der EDV wurden auch Büromöbel, Zeichengeräte und Organisationsmittel ausgestellt. In einer Zeit, in der Datenbankprogramme schon auf Heim-Computern laufen, scheint der gute alte Karteikasten immer noch nicht überholt. Mehrere Firmen präsentierten Neuentwicklungen.

Die Atmosphäre auf den Ständen der Bürodesigner und Büromöbel-Hersteller war beinahe klinisch steril. Wenn die dort gezeigten Einrichtungsvorschläge das Büro der Zukunft repräsentieren, so bedauere ich jeden dort Beschäftigten.

Kalte Farben, ein von extremer Funktionalität geprägtes Design und Materialien wie Kunststoff und Blech können mich jedenfalls nicht zu einer kreativen Arbeit inspirieren.



büro+ computer
München
4. bis 7. Mai 1988

Die meist regionalen Aussteller hatten viel Platz um sich zu präsentieren. In sechs großen Hallen verliefen sich die wenigen Besucher.

Dabei wurde gerade das Wort Kreativität auf dieser Messe stark strapaziert. Viele Aussteller waren bemüht, Kreativität in Zahlen und Prozentpunkten auszudrücken, um eine Steigerung derselben werbewirksam einzusetzen. Sätze wie: „Stellen Sie sich vor, die Kreativität Ihrer Mitarbeiter würde um fünf Prozent gesteigert", geisterten durch die Hallen. Es scheint, daß das unpopuläre Wort Produktivität durch diesen Modebegriff ersetzt wurde.

Auf dem Sektor der Zeichengeräte hält die Mikro-Elektronik Einzug. Es muß nicht gleich eine komplette CAD-Anlage erworben werden. Kleine Geräte, die, auf das Zeichenbrett aufgesetzt, Schriften und häufig benötigte Symbole zeichnen, werden immer beliebter.

Der Fachhandel

Die Fachhändler machten deutlich, daß der Anwender komplette Lösungen braucht und auch sucht. So versuchen sie, Hard- und Software im Paket zu vermarkten.

Dies ist ein sehr individuelles Geschäft. Standardpakete großer Softwarefirmen kommen hierbei selten zum Verkauf. Sie dienen höchstens als Basis für eine kundenspezifische Anpassung. Diese wird vom Fachhändler oder einem angeschlossenen Systemhaus ausgearbeitet.

Nach Aussage vieler Anbieter weiß der Kunde heute, bedingt durch gute Vorarbeit von Berufsverbänden, recht genau, was er möchte. Handwerker-Innungen und Berufsverbände haben Informationsmaterial erstellt, das es auch dem unbedarften Einsteiger ermöglicht, sich über die Einsatzgebiete und Grenzen der elektronischen Datenverarbeitung zu informieren. Auch Laien können somit bestimmen, wo sich der EDV-Einsatz für sie lohnt.

Einem kompetenten Kunden muß auch ein qualifizierter Fachhändler gegenüberstehen. Um dieser Forde-

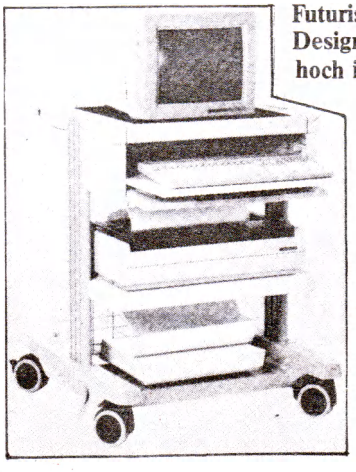
rung gerecht zu werden, ist der EDV-Fachhändler gut beraten, sich auf die Unterstützung weniger Branchen oder Anwendungen zu spezialisieren. Diese Tendenz ist bei allen größeren Systemhäusern zu erkennen.
Laut Münchner Messe AG nutzen derzeit nur ein Prozent aller deutschen Betriebsstätten die Möglichkeiten der elektronischen Datenverarbeitung. Selbst in Unternehmen mittlerer Größe, mit zwischen 30 und 500 Beschäftigten, liegt die Verbreitung von Computeranlagen nur bei 18 Prozent.

Software

Viele Selbständige, Freiberufler, Handwerker und Inhaber kleinerer Betriebe zögern bei einer derartigen Investition. Sie bemängeln am Computereinsatz ein Übermaß an unverständlicher Technik, ohne eine deutliche Steigerung der Produktivität als Gegenleistung zu erhalten. Ursache hierfür ist, nach beinahe einstimmiger Aussage von Anbietern und Berufsverbänden, der Software-Sektor.
Die in der Vergangenheit angebotene Software war zu abstrakt und stand zu nahe an der Technik des Computers. Effektiv einzusetzen war sie nur nach langer Einarbeitung durch technisch versierte Anwender. Erst eine Anpassung von Benutzeroberflächen und Programmen an die gewohnte Umgebung des Anwenders schafft Abhilfe.

Hardware

Ein wichtiger Aspekt bei der Beschaffung von Computer-Hardware sind die möglichen Erweiterungen. Besonders kleinere Unternehmen fürchten, von der rasanten Entwicklung der Computertechnik überrannt zu werden und verlangen Perspektiven für einen weiteren Ausbau ihrer Systeme.
Eine weitere Standardisierung von Hardware und Betriebssystemen hilft hier weiter. Die meisten derzeit verkauften Systeme entsprechen dem von IBM gesetzten Industriestandard. Die neue Produktline der IBM-PS/2-Modelle wird nur zögernd akzeptiert.
Als Betriebssystem wird derzeit überwiegend MS-DOS eingesetzt. Kleinere und mittlere Netzwerke laufen mit Novell-Software.
Ein Teil der Kunden hat schon etwas von dem Betriebssystem OS/2 gehört und will sich beim Hardwarekauf diese Option offenhalten. Viele Anbieter versprechen zwar auf OS/2 basierende Lösungen, präsentieren kann sie derzeit noch keiner.
Der Käufer vertraut in erster Linie großen Firmen. In der Hoffnung auf Kontinuität einer Produktlinie erwirbt er dabei bevorzugte Produkte der oberen Preisklasse, um nicht binnen kurzer Zeit in einer technologischen Sackgasse zu landen. Der Slogan des Handels heißt hier: „Lieber ein paar Mark zuviel investieren, als die gesamte Investition in den Sand setzen". Berufsverbände und Fachzeitschriften sind gefordert, diese Aussage zu relativieren.
Es ist sicher richtig, daß große etablierte Firmen eine kontinuierliche Weiterentwicklung ihrer Produktpalette garantieren. Es ist weiterhin richtig, daß die von diesen Firmen gebotene Unterstützung in Problemfällen zuverlässiger und schneller funktioniert als die von Billiganbietern aus Fernost.
Zwischen diesen beiden Extremen existiert aber eine große Zahl kleinerer Anbieter, die viel flexibler auf spezielle Wünsche eingehen können als Branchenriesen. Hier lassen sich bei gleichem Service bis zu 40 Prozent an Kosten sparen.
Die Möglichkeit, eine maßgeschneiderte Systemlösung zu erwerben, ist sicherlich ein Argument, auch ängstliche EDV-Einsteiger zu einem ersten Schritt zu ermuntern. Dieser Schritt sollte möglichst bald getan werden. Immer engere Märkte zwingen auch kleine Firmen zu einem Einsatz der EDV. Es muß nicht immer gleich eine umfassende Gesamtlösung sein, modulare Systeme gestatten es, sowohl die Hard- als auch die Software Stück für Stück zu erwerben und einzusetzen.
Dies hilft auch den Angestellten, sich an die Neuerungen an ihrem Arbeitsplatz langsam zu gewöhnen. Eine Steigerung der Produktivität ist nur dann zu erwarten, wenn die mit dem Medium Computer konfrontierten Personen das neue Arbeitsmittel auch akzeptieren.
Um diesen Schritt zu erleichtern, bot diese Messe ein Forum an. Fachreferenten sprachen dort über Themen im Umfeld der EDV. Dieser erstmals angebotene Service erwies sich als Flop. Die räumlichen und technischen Voraussetzungen waren zwar gut, die Qualität der Referate war es aber zum Großteil nicht. Von einem interessanten Themenkatalog angelockt, verließen daher viele Besucher die Vorträge frustriert nach kurzer Zeit des Zuhörens.
Von einem eifrigen, wenig sachkundigen Conferencier ermuntert, vergaßen die Referenten schnell ihr sicherlich vorhandenes Fachwissen und betrieben lautstarke Firmenwerbung.
Ich habe mir an einem Nachmittag drei Vorträge des Vertreters eines großen deutschen PC-Herstellers angehört. Er sollte zu verschiedenen Themen jeweils eine Stunde reden. Resultat waren jeweils 15 Minuten Firmenwerbung ohne Bezug zur angekündigten Thematik. Selbst der unerfahrene EDV-Anwender kann mit solch subjektiven Informationen nichts anfangen. Der Messeleitung sei geraten, dieses im Ansatz sicherlich positive Forum zu überdenken und in anderer Form zu realisieren.



Futuristisches Design steht hoch im Kurs.

Fazit

Eine regionale Messe wie Büro & Computer bietet einen interessanten Querschnitt geläufiger EDV-Anwendungen. Der direkte Kontakt zu einem Fachhändler oder Systemhaus der Region erleichtert eine weitere Beratung. Informationen von Berufsverbänden und Handwerkskammern vereinfachen potentiellen EDV-Anwendern das Gespräch mit einem Fachhändler. Trotz anwenderfreundlicher Software und weitgehend standardisierter Hardware sind noch lange nicht alle Berührungsängste abgebaut. Ein Messe-Forum mit einer objektiven Beratung ist wünschenswert. Das präsentierte Angebot ist so breit gefächert, daß nahezu jeder Anwender seine Lösung finden müßte. Der Einstieg in die EDV sollte, um konkurrenzfähig zu bleiben, bei vielen kleinen und mittleren Betrieben nicht mehr lange hinausgezögert werden. *LS*

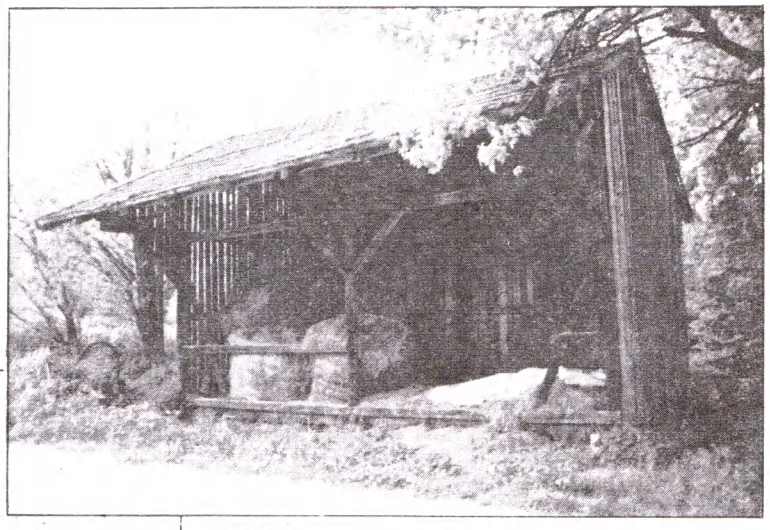
Speicher-Chips

Unter Dach und Fach

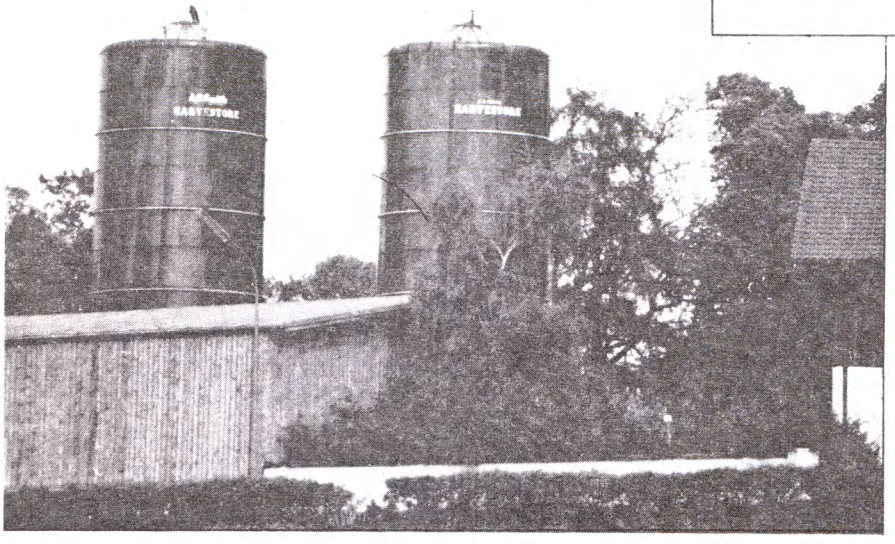
Wer mit Computern arbeitet, muß Daten und Programme speichern können. Externe Speicher kennt jeder: Diskette, Magnetband und Plattenspeicher. Aber auch der Rechner selbst braucht Speichermöglichkeiten: Arbeitsspeicher, Zwischenspeicher, Festwertspeicher und andere. Diese auf der Platine des Rechners untergebrachten Speicher sind alle in Chips untergebracht.

Entwicklungsgeschichte der Massenspeicher

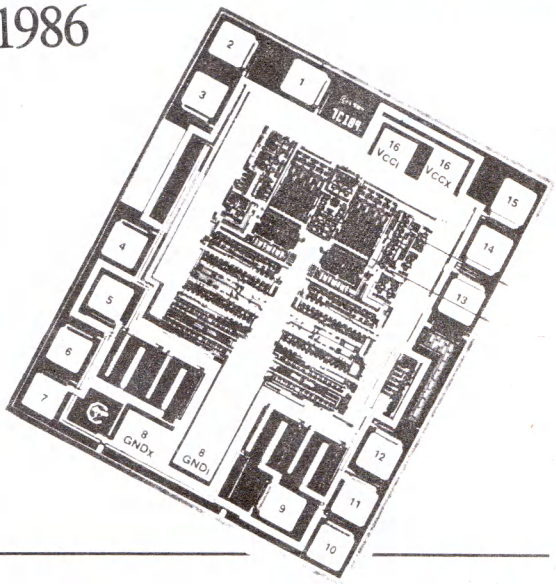
1860



1965



1986



Unter Dach und Fach

Programme und Daten werden in binär kodierter Form abgespeichert. Jeweils acht Bit werden zu einem Speicherplatz zusammengefaßt und erhalten eine Adresse. Eine spezielle Adressierlogik macht es möglich, jeden Speicherplatz einzeln anzusteuern. Dabei ordnet der Speicher einer vorgegebenen Adresse das jeweilige Datenwort (acht Bit) zu.
Die Anzahl der Gesamtbit eines Speichers ist immer eine Potenz von 2.

2^{10} Bit = 1024 Bit = 1 KBit
2^{11} Bit = 2048 Bit = 2 KBit
2^{15} Bit = 32768 Bit = 32 KBit
2^{16} Bit = 64536 Bit = 64 KBit

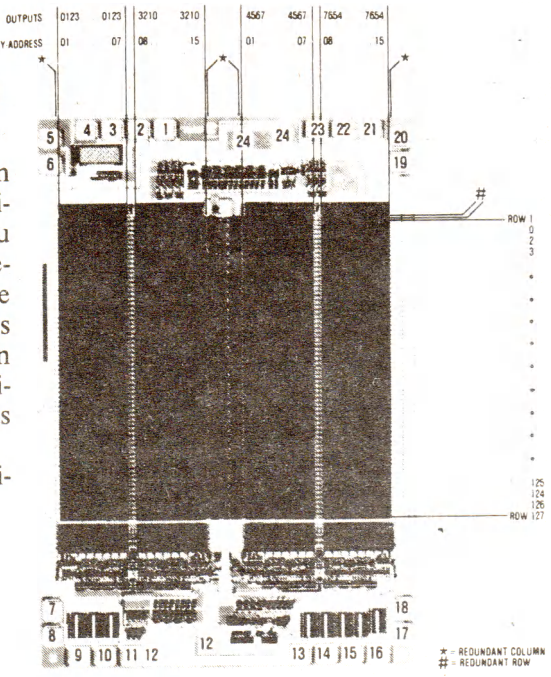
Bekanntlich ist das Bit (Binary Digit) die kleinste Darstellungseinheit für binäre Daten (0 oder 1). Um eine größere Anzahl von alphanumerischen Zeichen, etwa das Alphabet und die Ziffern von 0 bis 9, sowie Sonder- und Steuerzeichen darstellen zu können, werden jeweils acht Bit zu einer Einheit zusammengefaßt: zu einem Byte. Ein Byte ermöglicht so die Darstellung von 256 verschiedenen Zeichen. 256 ist die Anzahl der möglichen Kombinationen von 0 und 1, wenn insgesamt acht Stellen zur Verfügung stehen.
In der Praxis wird zu den acht Bit noch ein sogenanntes Prüfbit „gepackt" (sodaß es eigentlich neun Bit sind).
Die acht Datenbit können auf drei unterschiedliche Arten belegt werden:

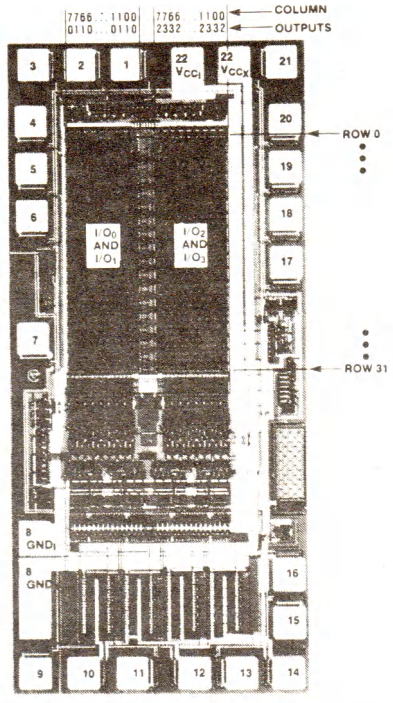
Binäre Darstellung.
Hier können Dezimalzahlen von 0 bis 255 (=256 Stück) in einem Byte gespeichert werden.
Dies ist im allgemeinen die günstigste Darstellungsart und bedingt die günstigste Speicheraufteilung.

Numerische Darstellung.
Hier wird ein Byte in zwei Tetraden unterteilt. Im sogenannten „gepackten Format" lassen sich so zwei Dezimalstellen verschlüsseln.

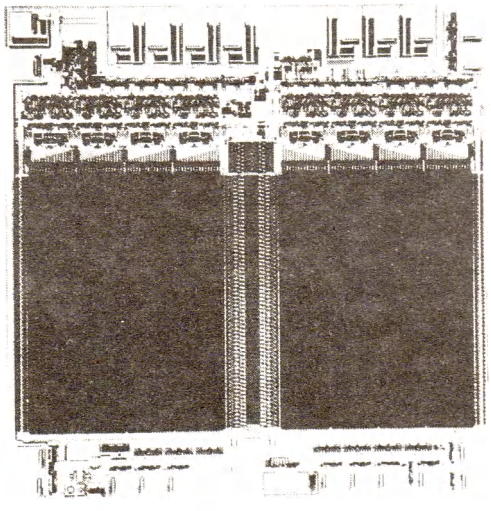
Alphanumerische Darstellung.
Wie bei der numerischen Darstellung wird auch hier das Byte in zwei Zonen von je vier Bit unterteilt: Die rechte Hälfte des Byte ist das niedrigwertige Bit und die linke Hälfte das höherwertige Bit. Dezimalziffern werden nur in den niedrigwertigen Bit verschlüsselt, während Buchstaben und Sonderzeichen durch die Kombination einer Ziffernverschlüsselung (s. o.) und einer Zonenverschlüsselung in den übrigen Bit vorgenommen wird. Speichergrößen werden daher nicht in Bit, sondern in Byte angegeben:

2^{19} Bit = 524.288 Bit = 512 KBit = 64 KByte
2^{20} Bit = 1.048.576 Bit = 1.024 KBit = 128 KByte
2^{21} Bit = 2.097.152 Bit = 2.048 KBit = 256 KByte
2^{22} Bit = 4.194.304 Bit = 4.096 KBit = 512 KByte

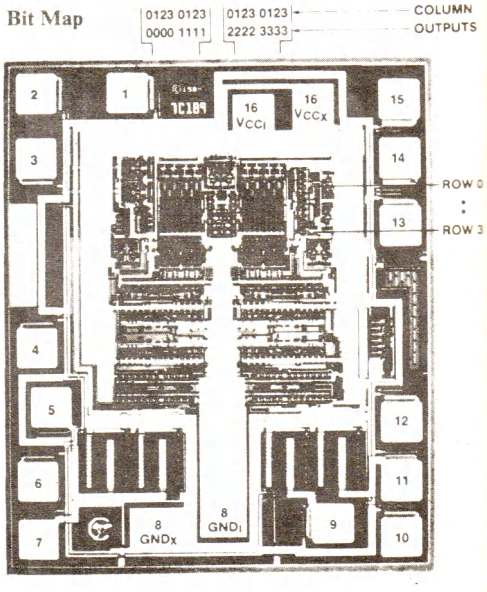




Das Layout der verschiedenen Speicherchips ist sehr ähnlich. Die verschiedenen Technologien sind nicht zu unterscheiden.



Bit Map



tausende von Speicherzellen.

RAM

Beim RAM (Random Access Memory) handelt es sich um einen Speicher mit wahlfreiem Zugriff. Er kann wahlweise ausgelesen oder mit neuen Daten beschickt werden. Aus diesem Grunde spricht man auch von einem Schreib- und Lesespeicher. Wahlfrei bedeutet aber auch, daß man sich Daten (über Adressen) auswählen kann, im Gegensatz zu einem Speicher, der nur den kompletten Inhalt freigibt. RAM-Speicher ermöglichen einen direkten Zugriff durch direkte Adressierung.
Eine weitere Eigenschaft zeichnet das

RAM aus. Nach dem Einschalten des Rechners enthält der RAM-Speicher irgendeine zufällige Information, also nicht nur Nullen. Das resultiert aus der Flüchtigkeit dieses Speichermediums: Beim Abschalten des Rechners verliert er die gespeicherten Informationen und befindet sich somit in einem undefinierten Zustand, man kann also auch nicht sagen, daß er keine Informationen enthält, eben nur ungeordnete.
Das RAM benötigt zum Erhalten der auf ihm gespeicherten Information elektrische Energie. Durch die Entwicklung extrem stromsparender RAM-Bausteine wurde es möglich, diese bei ausgeschaltetem Rechner mit Batteriestrom zu versorgen. Dies bedeutet, die Information bleibt erhalten, solange der Ladezustand der Batterie es erlaubt. Bei erneutem Einschalten des Rechners wird die Batterie nachgeladen. Man spricht hier von Batteriepufferung.

ROM

Wie der Name ROM (für Read Only Memory) schon sagt, können hier nur Daten gelesen werden. Weil der Informationsinhalt des ROM nicht verändert werden kann, spricht man auch von einem Festwertspeicher. ROM-Chips haben eine wichtige Funktion im Rechner: Sie enthalten beispielsweise Betriebssystem, Programmiersprachen oder eine Laderoutine (für das Betriebssystem auf der Diskette und anderes). Beim Einschalten des Rechners wird ein Programm von einem ROM gebootet (also automatisch geladen). Dieses Programm ermöglicht erst den Dialog mit dem Computer (über Tastatur, Bildschirm). Meist ist dieses Programm entweder ein Lader für das Betriebssystem oder das Betriebssystem selbst.
Der Inhalt des ROM ist unveränderbar und wird auch beim Ausschalten des Rechners nicht gelöscht. Zum Erhalten der gespeicherten Information ist keine Energie notwendig.
Festwertspeicher sind wegen der nicht notwendigen Adressierung sehr schnell und leicht in Systeme zu integrieren.

PROM

Das PROM (Programmable Read Only Memory) kann vom Anwender nur einmal programmiert werden. Die Informationen werden eingebrannt und können dann nicht mehr geändert werden. Dieser programmierbare Festwertspeicher verhält sich daraufhin genau wie ein ROM.
Ob ein ROM oder ein PROM in ein Gerät eingebaut wird, hängt im wesentlichen von der produzierten Stückzahl des Gerätes ab. Bei großen Serienfertigungen wird man meist ROMs verwenden, deren Entwicklung zwar sehr teuer ist, die Massenproduktion aber billig wird. PROMs müssen einzeln programmiert werden, daher eignen sie sich mehr für kleine Serien oder sogar Einzelanfertigungen.

EPROM

Wie das PROM kann auch das EPROM (Erasable Programmable Read Only Memory) vom Anwender programmiert und dann wie ein ROM benutzt werden. Das EPROM ist ein löschbarer Speicher. Mit Hilfe von UV-Licht kann eine Programmierung rückgängig gemacht, das EPROM gelöscht werden. Daraufhin ist es wieder bereit, neue Informationen aufzunehmen und zu speichern. Daher eignen sich EEROMs hervorragend für Software-Entwicklungen. Meist wird es nicht fest auf Platinen verlötet, sondern nur in IC-Sockel gesteckt, um es zum Löschen und erneuten Programmieren von der Platine wieder entfernen zu können.
Der EPROM-Chip besitzt auf seiner Oberfläche ein Fenster für den Löschvorgang. Dies unterscheidet ihn schon rein äußerlich von anderen Chips.
Zum Löschen des Speicherinhaltes wird der Chip unter eine spezielle Löschlampe (UV-Licht) gebracht.

EEPROM

EEPROM (Electrical Erasable and Program Read Only Memory) ist eine Weiterentwicklung der zuvor beschriebenen EPROM-Chips. Im Gegensatz zu diesen müssen sie nicht mit UV-Licht gelöscht werden, um neu programmiert werden zu können. Einzelne Speicherzellen können bei erneuter Programmierung einfach überschrieben werden. Sie sind bedienungsfreundlicher als EPROMs und zeichnen sich zudem durch eine höhere Lebensdauer aus, durch den Wegfall der möglichen UV-Bestrahlung.

Dipl. Ing. (FH) Oliver Rosenbaum

Gewusst wie!

Immer wieder sind es gerade die kleinen Helferlein, die das Herz des geplagten Anwenders höher schlagen lassen

1. Ein Druckertreiber für 24 Nadeldrucker

Die 24 Nadeltechnik ist zur Zeit der Renner auf dem Druckermarkt. Damit der User die Möglichkeiten seines 24 Nadeldruckers auch richtig nutzen kann, präsentieren wir einen Treiber für einen solchen Drucker.

2. Ein Variabellister sorgt für Durchblick

Es ist immer gut zu wissen, welche Variablen-Namen in einem GFA-Basic Programm vorkommen. Unser Programm "Varprint" zeigt nun alle in einem Programm vorkommenden Variablen am Bildschirm. Für Basic-Programmierer, die leicht den Überblick über die im Programm verwendeten Variablen verlieren, ein idealer Helfer in der Not.

```
' Druckertreiber für 24 Nadeldrucker
'
'
Lprint Chr$(27)+"@"
Mg$=""+Chr$(27)+"*"+Chr$(38)+Chr$(128)+Chr$(2)
Open "",#99,"LST:"
For Z=0 To 399 Step 24
  A$=""
  For S=0 To 639
    For I=0 To 7
      If Point(S,Z+I)=1
        A1=A1+2^(7-I)
      Endif
    Next I
    For I=0 To 7
      If Point(S,8+Z+I)=1
        A2=A2+2^(7-I)
      Endif
    Next I
    For I=0 To 7
      If Point(S,16+Z+I)=1
        A3=A3+2^(7-I)
      Endif
    Next I
    A$=A$+Chr$(A1)+Chr$(A2)+Chr$(A3)
    A1=0
    A2=0
    A3=0
  Next S
  Print #99,Mg$;A$;Chr$(13);
  Print #99,Chr$(27);"3";Chr$(24)
Next Z
Close #99
```

```
' Varprint
'
'
Fileselect "a:\*.bas","",A$
Open "I",#1,A$
For T=0 To 127
  Void Inp(#1)
Next T
Print "  Variablenliste von ";A$
Print String$(80,"-")
Do
  C$=""
  Repeat
    Inc Z
    A=Inp(#1)
    C$=C$+Chr$(A)
  Until A<32
  C$=Left$(C$,Len(C$)-1)
  Print Chr$(9);C$;"  ";
  Exit If Instr(C$,"*")<>0
Loop
```

Diskmon 1.0

Der Bildschirm ist bei Diskmon 1.0 in drei Teile gegliedert:
1. Statusfenster
2. Anzeige und Editierfenster
3. Menüfenster

Zu 1.

Im Statusfenster werden aktuelle Werte angezeigt:
- Track: die Spur
- Sektor: der Sektor
- Side: die selektierte Diskseite, bei SF 354 nur „0" möglich
- Drive: das Laufwerk das angesprochen werden soll
- Status: bei „0" ist beim letzten Schreibvorgang kein Fehler aufgetreten, andernfalls steht hier dann eine negative Fehlernummer
- Byte: das Byte auf dem sich der Cursor gerade befindet

Zu 2.

Im Editierfenster wird der aktuelle Sektorinhalt angezeigt. Da nur 256 Bytes auf einmal angezeigt werden. Durch das Betätigen der linken Maustaste kann so weiter gescrollt werden. Mit dem Mauszeiger kann innerhalb dieses Fensters der Cursor auf ein beliebiges Byte im HEX- oder ASCII-Format plaziert werden. Wird ein ASCII-Byte verändert, so wird auch gleichzeitig der HEX-Code verändert und umgekehrt. Im Statusfenster wird jetzt angezeigt, welches Byte gerade verändert wird. Dies alles funktioniert aber nur, wenn sich der Cursor in diesem Fenster befindet.

Zu 3.

Im Menüfenster können die verschiedenen Funktionen des Diskmons ausgewählt werden:
- Track + der aktuelle Track wird erhöht
- Track – der aktuelle Track wird verringert
- Sektor + siehe Track +
- Sektor – siehe Track –
- Write Sektor auf den aktuellen Sektor und Track wird der Buffer zurückgeschrieben
- Side 0 Seite 0 wird selektiert
- Side 1 Seite 1 wird selektiert
- Boot Sektor. Der Track wird auf 0 und der Sektor auf 1 zurückgesetzt
- Set T&S Der Track und Sektor können per Tastatur eingegeben werden
- Drive Das Laufwerk wird ausgewählt
- Format Track. Tracks können formatiert werden
- Print Sektor. Der aktuelle Sektor wird ausgedruckt
- Exit Das Programm wird verlassen und kehrt zum Desktop zurück.

```
'
' DISKMONITOR
'
'
' variablen
Dim Bu$(32)
Adr=1
Fi=&HE5
Dev=0
Sec=1
Tra=0
Tramax=82
Sécmax=9
Sid=0
Fehl=0
Res=1
Buf$=Space$(512)
Sbuf$=Space$(512*Secmax)
Nbuf$=Space$(512*Secmax)
@Read(Tra,Sec)
@Screen
@Wandle
@Update
@Asciihex
Print Chr$(27)+"e";
Print At(56,5);
Do
  Let Out=0
  Defmouse 0
  Do
    @Abfrage
    If Selected<>0
      On Selected Gosub Inctra,Dectra,
          Incsec,Decsec,Wrtsec,Side0,Side1,
            Bootsec,Trasec,Ch_drive,
              Suchen,Format,Print,Exit
      Selected=0
    Endif
    Exit If Out=1
  Loop
  Let Out=0
  Defmouse 1
  Do
    @Edit
    Exit If Out=1
  Loop
Loop
Procedure Inctra
  If Tra<=Tramax
    Inc Tra
    @Read(Tra,Sec)
    @Asciihex
    @Update
  Endif
Return
Procedure Dectra
  If Tra>0
    Dec Tra
    @Read(Tra,Sec)
    @Asciihex
```

```
      @Update
    Endif
  Return
  Procedure Incsec
    If Sec<=Secmax
      Inc Sec
      @Read(Tra,Sec)
      @Asciihex
      @Update
    Endif
  Return
  Procedure Decsec
    If Sec>1
      Dec Sec
      @Read(Tra,Sec)
      @Asciihex
      @Update
    Endif
  Return
  Procedure Side0
    Sid=0
    @Read(Tra,Sec)
    @Asciihex
    @Update
  Return
  Procedure Side1
    Sid=1
    @Read(Tra,Sec)
    @Asciihex
    @Update
  Return
  Procedure Bootsec
    Sid=0
    Tra=0
    Sec=1
    @Read(Tra,Sec)
    @Asciihex
    @Update
  Return
  Procedure Trasec
    Get 100,150,525,255,Rette$
    Deftext 1,0,0,13
    @Window(100,150,420,100,"Track & Sektor",1)
    Print At(25,13);Chr$(27)+"f";"Track :";
    Form Input 2 As Tr$
    Print At(25,14);"Sektor:";
    Form Input 2 As Se$
    If Val(Se$)<Secmax And Val(Tr$)<Tramax
      Tra=Val(Tr$)
      Sec=Val(Se$)
      @Read(Tra,Sec)
      @Asciihex
      @Update
    Endif
  Return
  Procedure Suchen
    Get 100,150,525,255,Rette$
    Deftext 1,0,0,13
    @Window(100,150,420,100,"Suchen",1)
    Print At(25,13);Chr$(27)+"f";"Such-String:";
```

```
  Form Input 15 As Su$
  For T=0 To Tramax
    Print At(5,8);T;
    Fehl=Xbios(8,L:Varptr(Sbuf$),
       L:Fi,Dev,1,T,Sid,Secmax)
    Su=Instr(Sbuf$,Su$)
    If Su<>0
      Print At(25,15);"Track:"+Str$(T)+"
         Sector:";Trunc(Su/512)
      @Read(Tra,Trunc(Su/512)+1)
      @Asciihex
      Tra=T
      Sec=Trunc(Su/512)+1
      T=Tramax
    Endif
  Next T
  Put 100,150,Rette$
  Print Chr$(27)+"e";
Return
Procedure Wrtsec
  Alert 1," Wollen Sie wirklich !
     Track :"+Str$(Tra)+" !
       Sector:"+Str$(Sec)+"  !
        auf Disk zurückschreiben ?",1,"  JA  !
          NEIN ",Al
  If Al=1
    @Write(Tra,Sec)
  Endif
Return
Procedure Format
  Get 100,150,525,255,Rette$
  Deftext 1,0,0,13
  @Window(100,150,420,100,"Format Track",1)
  Print At(25,12);Chr$(27)+"f";"Track    : ";
  Form Input 2 As Tr$
  Print At(40,12);"bis Track : ";
  Form Input 2 As Btr$
  Print At(25,13);"Sectoren : ";
  Form Input 2 As Sc$
  Print At(25,14);"Side     : ";
  Form Input 2 As Si$
  Alert 1,"Wirklich formatieren ??",2,"  JA  !
     NEIN ",Al
  If Al=1
    Defmouse 2
    For Tra=Val(Tr$) To Val(Btr$)
      Fehl=Xbios(10,L:Varptr(Nbuf$),L:0,Dev,
         Val(Sc$),Tra,Sid,1,L:&H87654321,&HE5E5)
      Print At(35,15);"Track#";Tra
    Next Tra
    Defmouse 0
  Endif
  Print Chr$(27)+"e";
  Put 100,150,Rette$
Return
Procedure Edit
  Let Out=0
  Do
    Print At(56,5);Chr$(27)+"e";
    Cx=56
```

```
Cy=5
Flag=1
Repeat
  Key$=Mid$(Inkey$,1,1)
  Key2$=Mid$(Inkey$,2,1)
  Mouse Mx,My,Mk
  If Mk=2
    Defmouse 2
    If Sx>0
      Sub Sx,16
      Adr=1
    Else
      Adr=257
      Add Sx,16
    Endif
    @Update
    @Wandle
    For T=0 To 15
      Print At(6,5+T);Bu$(T+Sx)
    Next T
    @Mkey(0)
    Print At(56,5);
    Defmouse 1
  Endif
  '
  ' HEX feld
  '
  For T=0 To 52 Step 3
    If Int(Mx/8+1)=T
      Flag1=1
      T=52
    Else
      Flag1=0
    Endif
  Next T
  If (Mk=1 And Mx>8*5 And Mx<8*52
    And My>16*4 And My<16*20) And Flag1=1
    Cx=Int(Mx/8+1)
    Cy=Int(My/16+1)
    Mkx=Cx
    Adr=((Mkx-6)/3)+((Sx+Cy-5)*16)+1
    Print At(70,2);Adr;
    Print At(Cx,Cy);
    Zae=0
    Repeat
    Until Mousek=0
    Asciihex=2
  Endif
  If (Key$=>"0" And Key$<="9") Or
   (Key$=>"A" And Key$<="F") Or
      (Key$=>"a" And Key$<="f")
        And Asciihex=2
    Key$=Upper$(Key$)
    Print At(Cx,Cy);Key$;
    Inc Cx
    A=Trunc(Cx/3)+54
    Adr=((Mkx-6)/3)+((Sx+Cy-5)*16)+1
    Print At(70,2);Adr;
    Mid$(Bu$(Sx+Cy-5),Mkx+Zae,1)=Key$
    A$=Mid$(Bu$(Sx+Cy-5),Mkx,2)
```

```
      @Wandlehex(A$)
      If Dez>32 And Dez<128
        Print At(A+B,Cy);Chr$(Dez);
      Else
        Print At(A+B,Cy);".";
      Endif
      Print At(Cx,Cy);
      Mid$(Buf$,Adr,1)=Chr$(Dez)
      Inc Zae
      If Zae=2
        Add Mkx,3
        Zae=0
        Inc Cx
        If Cx>52
          B=0
          Cx=6
          Mkx=6
          If Cy<20
            Inc Cy
          Endif
          Print At(Cx,Cy);
        Endif
      Endif
    Endif
    '
    ' ASCII feld
    '
    If Mk=1 And Mx>8*55 And Mx<8*71
      And My>16*4 And My<16*20
      Cx=Int(Mx/8+1)
      Cy=Int(My/16+1)
      Adr=(Cx-55)+((Sx+Cy-5)*16)
      Print At(70,2);"    ";Chr$(8);
       Chr$(8);Chr$(8);Adr
      Print At(Cx,Cy);
      Repeat
      Until Mousek=0
      Asciihex=1
    Endif
    If Asc(Key$)=>32 And Cx>=56 And Asciihex=1
      Print At((Cx-56)*3+5,Cy);" ";Hex$(Asc(Key$));
      Adr=(Cx-55)+((Sx+Cy-5)*16)
      Mid$(Buf$,Adr,1)=Key$
      Print At(70,2);"    ";Chr$(8);Chr$(8);Chr$(8);Adr;
      Print At(Cx,Cy);Key$;
      Inc Cx
      If Int(Cx)=72
        Cx=56
        If Cy<20
          Inc Cy
        Endif
        Print At(Cx,Cy);
      Endif
    Endif
    If My>330
      Let Out=1
    Endif
  Until Out=1
  Exit If Out=1
Loop
```

```
Return
Procedure Wandle
  For T=1 To Len(Buf$) Step 16
    Pr1$=""
    Pr2$=""
    For X=0 To 15
      Mi$=Mid$(Buf$,T+X,1)
      H$=Hex$(Asc(Mi$))+" "
      If Len(H$)=2
        Pr1$=Pr1$+"0"+H$
      Else
        Pr1$=Pr1$+H$
      Endif
      If Asc(Mi$)=>32 And Asc(Mi$)<=128
        Pr2$=Pr2$+Mi$
      Else
        Pr2$=Pr2$+"."
      Endif
    Next X
    Bu$(T/16)=Pr1$+"  "+Pr2$
  Next T
Return
'
Procedure Read(Tra,Sec)
  Defmouse 2
  Fehl=Xbios(8,L:Varptr(Buf$),
   L:Fi,Dev,Sec,Tra,Sid,1)
Return
'
Procedure Write(Tra,Sec)
  Defmouse 2
  Fehl=Xbios(9,L:Varptr(Buf$),L:
   Fi,Dev,Sec,Tra,Sid,1)
  Defmouse 0
Return
'
Procedure Print
  Alert 1,"Wirklich drucken ??",1,"  JA  |
    NEIN ",Al
  If Al=1
    Lprint "Track:"+Str$(Tra)+"   Sektor:"+Str$(Sec)
    Lprint String$(80,"-")
    For T=1 To 32
      Lprint Space$(Tab);Bu$(T)
    Next T
  Endif
Return
'
Procedure Window(X,Y,H,B,Win$,Flag)
  Y=Y/Res
  B=B/Res
  If Flag=1
    Dpoke Gintin,320
    Dpoke Gintin+2,200/Res
    Dpoke Gintin+4,50
    Dpoke Gintin+6,50/Res
    Dpoke Gintin+8,X
    Dpoke Gintin+10,Y
    Dpoke Gintin+12,H
    Dpoke Gintin+14,B
```

```
    Gemsys 73
  Endif
  Graphmode 1
  Pbox X,Y,H+X,B+Y
  Graphmode 3
  Pbox X,Y,H+X,B+Y
  Graphmode 1
  Box X+2,Y+2,H-2+X,B-2+Y
  Pbox X+5,Y+B,X+H,(Y+B)+5/Res
  Pbox X+H,Y+5,H+5+X,(B+Y)+5/Res
  Text X+(H-(Len(Win$)*16))/2,Y+20,Win$
Return
Procedure Screen
  Restore
  Deffill 1,1,1
  Pbox 0,0,639,399
  Deffill 1,2,4
  Prbox 0,0,639,399
  Deffill 1,1,1
  @Window(25,55,585,275,"",1)
  @Window(25,5,585,40,"",1)
  @Window(25,340,585,50,"",1)
  Deftext 1,0,0,4
  For T=1 To 14
    Read Te1$,Te2$
    Box (40*T),350,(40*T)+35,380
    Text (40*T)+3,360,Te1$
    Text (40*T)+3,370,Te2$
  Next T
  Deftext 1,0,2700,13
  Graphmode 2
  Text 619,40,"Diskmon 1.0 written 1987
   by Jan Kubuschok"
Return
Procedure Update
  Print At(5,2);Space$(70);
  Print At(5,2);"Track:  ";Chr$(8);Chr$(8);Tra
  Print At(15,2);"Sector:";Sec
  Print At(26,2);"Side:";Sid
  Print At(36,2);"Drive:";Dev
  Print At(46,2);"Status:";Fehl
  Print At(65,2);"Byte:";Adr
  Print At(56,5);
Return
Procedure Abfrage
  Mouse X,Y,K
  If Y<320
    Let Out=1
  Endif
  For T=1 To 14
    If X>(40*T) And Y>350 And X<(40*T)+35
     And Y<380 And K=1
      Graphmode 3
      Pbox (40*T),350,(40*T)+35,380
      @Mkey(0)
      Pbox (40*T),350,(40*T)+35,380
      Selected=T
      Graphmode 1
    Endif
  Next T
```

```
Return
Procedure Mkey(Flag)
  Repeat
  Until Mousek=Flag
Return
Procedure Asciihex
  Sx=0
  @Wandle
  For T=0 To 15
    Print At(6,5+T);Bu$(T+Sx)
  Next T
  Defmouse 0
Return
Procedure Ch_drive
  Flg=0
  Deftext 1,0,0,13
  Graphmode 1
  Dr=Bios(10)
  A$=Bin$(Dr)
  Z=(640-(Len(A$)-1)*30)/2
  Get Z-20-2,120-2,Z+((Len(A$))*30)+22,257,Rette$
  Deffill 1,0,0
  Pbox Z-20-2,120-2,Z+((Len(A$))*30)+22,257
  Deffill 1,1,1
  Text (Z+Len(A$)*30/2)-((Len("Drive")*8)/2),
    160,"Drive"
  Text (Z+Len(A$)*30/2)-((Len("Drivet")*8)/2),140,
    "Select"
  Box Z-20,120,Z+((Len(A$))*30)+20,255
  Box Z+(Len(A$)*30/2)-30,220,Z+60+
   (Len(A$)*30/2)-30,240
  Box Z+(Len(A$)*30/2)-30+1,220+1,Z+60+
    (Len(A$)*30/2)-30-1,240-1
  Text (Z+Len(A$)*30/2)-((Len(" OK ")*8)/2),236," OK "
  For T=0 To Len(A$)-1
    B$=Mid$(A$,T,1)
    If B$="1"
      Box Z+(30*T),175,Z+25+(30*T),200
      Text Z+8+(30*T),194,Chr$(65+T)
    Else
      Graphmode 3
      Deffill 1,3,1
      Pbox Z+(30*T),175,Z+25+(30*T),200
      Text Z+8+(30*T),194,Chr$(65+T)
      Graphmode 1
    Endif
  Next T
  Deffill 1,1,1
  Graphmode 3
  T=Gemdos(&H19)
  Pbox Z+(30*T),175,Z+25+(30*T),200
  Do
    Mouse X,Y,K
    If K=1
      Mouse X,Y,K
      For T=0 To Len(A$)-1
        If X>Z+(30*T) And X<Z+25+(30*T)
          And Y>175 And Y<200
           And Mid$(A$,T,1)="1"
          Pbox Z+(30*T),175,Z+25+(30*T),200
```

```
            @Mkey(0)
            T1=T
            T=Gemdos(&H19)
            Pbox Z+(30*T),175,Z+25+(30*T),200
            Dev=T1
            Void Gemdos(&HE,T1)
          Endif
        Next T
        If X>Z+(Len(A$)*30/2)-30+1
          And Y>220+1 And X<Z+60+(Len(A$)*30/2)-30-1
           And Y<240-1
          Graphmode 3
          Pbox Z+(Len(A$)*30/2)-30+1,
            220+1,Z+60+(Len(A$)*30/2)-30-1,240-1
          @Mkey(0)
          Pbox Z+(Len(A$)*30/2)-30+1,
           220+1,Z+60+(Len(A$)*30/2)-30-1,240-1
          Flg=1
        Endif
      Endif
      Exit If Flg=1
    Loop
    @Update
    Put Z-22,118,Rette$
Return
Procedure Wandlehex(W$)
    Local A,T,Z
    Dez=0
    Z=0
    For T=1 To 0 Step -1
      Inc Z
      A=Asc(Mid$(W$,Z,1))
      If A>47 And A<58
        Dez=Dez+Val(Mid$(W$,Z,1))*16^T
      Else
        Dez=Dez+(Asc(Mid$(W$,Z,1))-55)*16^T
      Endif
    Next T
Return
Procedure Exit
    Alert 1," Abbruch ?? ",1,"  JA  ! NEIN ",A1
    If A1=1
      End
    Endif
Return
Data Track,  +  ,Track,  -  , Sek ,
 +  , Sek ,  -  ,Write, Sec ,Side,
  0  ,Side,  1  ,Boot, Sek , Set,
   T&S,Drive,,Su-,chen,
     Format,Track,Print, Sec, Exit,
```

Vokabel-Champ 1.0

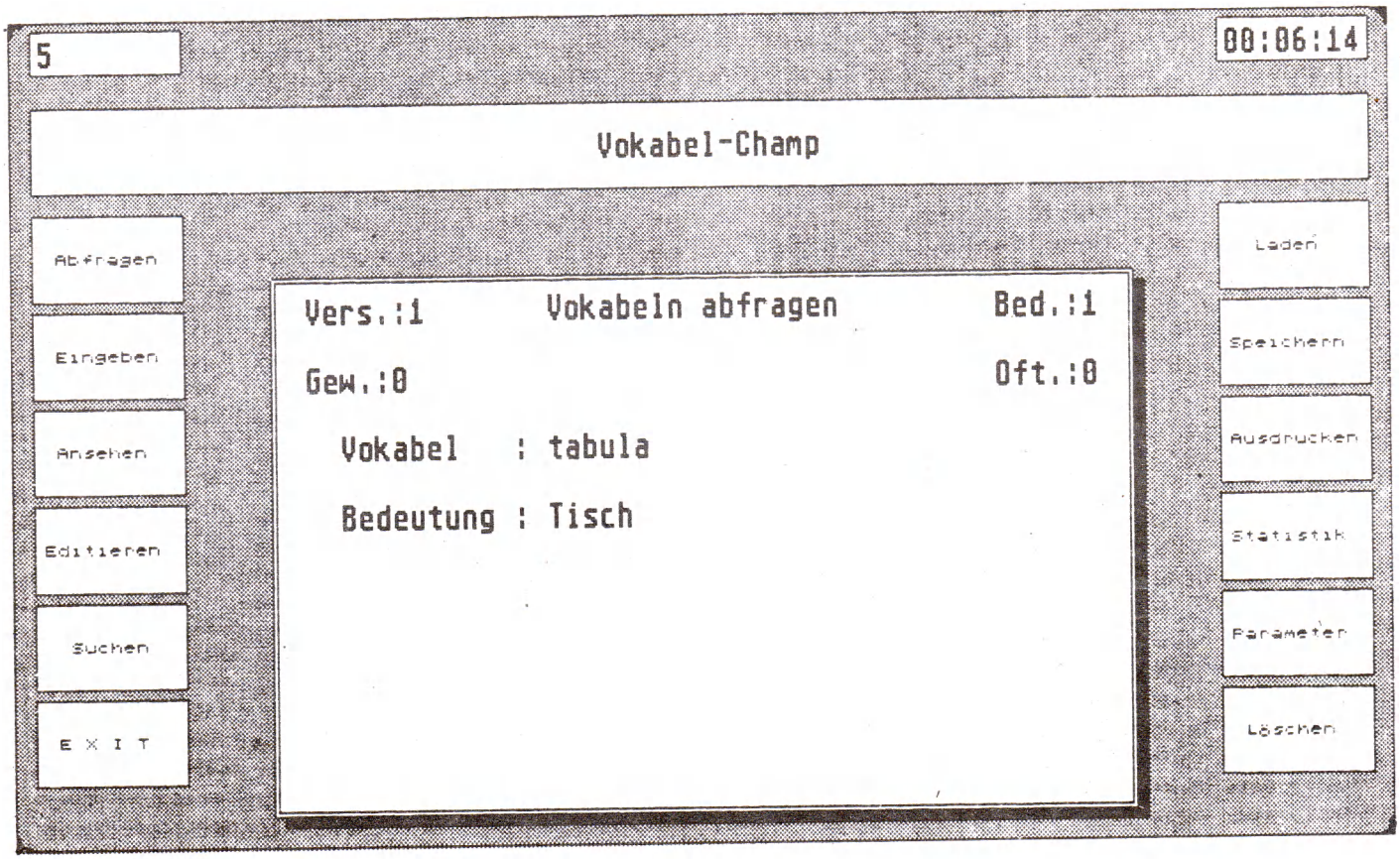
MENÜPUNKTE:

– Abfragen
(Vokabel-Bedeutung)
(Bedeutung-Vokabel)
Mit der Variablen OFT kann eingestellt werden, wie oft eine Vokabel richtig (in allen Bedeutungen) gewußt werden muß.
(Siehe auch Parameter)
Im Fenster steht dann:
– Vers. für Anzahl der Versuche
– Bed. für Anzahl der Bedeutungen
– Gew.für Anzahl der richtigen Eingaben
– Oft. für wie oft die Vok. schon gewußt wurde
Mit UNDO kann dieser Menüpunkt abgebrochen werden.

– Editieren
Hier können die Vokabeln verbessert werden.
Zunächst fragt das Programm nach der zu verbessernden Vokabel.
Alles weitere läuft wie bei der Vokabeleingabe.

– Suchen
Hier kann nach der Bedeutung oder nach der Vokabel gesucht werden.
Prinzipiell alle Vokabeln, die mit der eingegebenen Zeichenkette beginnen. Es muß nicht auf Groß- und Kleinschreibung geachtet werden.

– Exit
Programm verlassen



– Eingeben
Hier können die Vokabeln eingegeben werden, wobei zuerst nach der Vokabel gefragt wird und dann nach den Bedeutungen. Es sind maximal fünf Bedeutungen möglich. Wird eine Bedeutung ohne eine Eingabe abgeschlossen (Return) geht das Programm davon aus, daß keine weiteren Bedeutungen eingegeben werden wollen.
Es erscheint dann eine kleine BOX und es kann nun die Eingabe bestätigt werden.
Mit UNDO kann dieser Menüpunkt abgebrochen werden.

– Ansehen
Es werden alle Vokabeln und deren Bedeutungen angezeigt.
Nach jeder Vokabel wartet das Programm auf eine der beiden Maus-Tasten.
Mit der linken Maustaste wird die nächste Vokabel angezeigt, mit der rechten kann dieser Menüpunkt verlassen werden.

– Laden
Vokabeln in den Speicher laden

– Speichern
Vokabeln im Speicher speichern

– Ausdrucken
Die Vokabeln werden auf dem Drucker ausgegeben.

– Statistik
Eine kleine Statistik wird angezeigt.

– Parameter
Es kann eingestellt werden, wie oft eine Vokabel gewußt werden muß.

– Löschen
Der Vokabelspeicher wird gelöscht.

```
'
'
'
'
On Error Gosub Fehler
Dim Vok$(1000,5)
Dim Vok(1000)
Revon$=Chr$(27)+"p"
Revoff$=Chr$(27)+"q"
Oft=3
@Screen
'
' main job
'
@Main
Procedure Main
  Selected=0
  Sput Screen$
  Do
    Repeat
      @Watch_menue
      @Clock
    Until Selected<>0
    On Selected-1 Gosub Abfragen,Eingeben,
          Ansehen,Editieren,Suchen,Exit,
              Load_all,Save_all,Drucken,
                   Statistik,Parameter,Losch
    @Mkey(0)
    Sput Screen$
    Selected=0
    Graphmode 3
  Loop
Return
'
' Vokabel Routinen :
'
' A B F R A G E N
'
Procedure Abfragen
  Arrayfill Vok(),0
  Gewusst=0
  @Pruef
  @Window(120,120,400,250,"Vokabeln abfragen",1)
  Alert 1,"Vokabel ==> Bedeutung |
     oder | Bedeutung ==>Vokabel ",1,"Vo > Be|Be > Vo",Al
           On Al Gosub Abfragen1,Abfragen2
Return
Procedure Abfragen1
  Zae=0
  Do
    Flag=0
    Repeat
      Zu=Random(Vok)
      If Vok(Zu)=Oft
        Flag=0
      Else
        Inc Zae
        Flag=1
      Endif
```

```
      Until Flag=1
      Graphmode 1
      Deffill 0
      Pbox 240,146/Res,450,350/Res
      Deffill 1
      Bedeut=0
      For T=1 To 5
        If Len(Vok$(Zu,T))>1
          Inc Bedeut
        Endif
      Next T
      Print At(18,9);"Vers.:";Zae
      Print At(18,11);"Gew.:";Gewusst
      Print At(58,11);"Oft.:";Vok(Zu)
      Print At(58,9);"Bed.:";Bedeut
      Print At(20,13);"Vokabel    : ";Vok$(Zu,0)
      For Be=1 To Bedeut
        Print At(20,15);"Bedeutung : ";
        @Input(32,15,20,"")
        Bedeutung$=Upper$(B$)
        Flag=0
        For T=1 To Bedeut
          If Bedeutung$=Upper$(Vok$(Zu,T))
             And Not Bedeutung$=""
            Flag=1
            Gflag=1
            Print At(32,16+T);Vok$(Zu,T)
          Endif
        Next T
        If Flag=0
          If Vok(Zu)<>0
            Dec Vok(Zu)
          Endif

          For T=1 To Bedeut
            Print Revon$;
            Print At(32,16+T);Vok$(Zu,T)
            +Space$(20-Len(Vok$(Zu,T)))
            Print Revoff$;
          Next T
          Be=Bedeut
        Else
          Inc Vok(Zu)
          If Vok(Zu)=Oft
            Inc Oftgew
          Endif
        Endif
      Next Be
      If Flag=1
        Gflag=0
        Inc Gewusst
      Endif
      Repeat
      Until Inkey$=Chr$(13)
      Exit If Oftgew=>Vok
    Loop
  Return
  Procedure Abfragen2
    Zae=0
    Do
      Flag=0
```

```
    Repeat
      Zu=Random(Vok)
      If Vok(Zu)=Oft
        Flag=0
      Else
        Inc Zae
        Flag=1
      Endif
    Until Flag=1
    Bedeut=0
    For T=1 To 5
      If Len(Vok$(Zu,T))>1
        Inc Bedeut
      Endif
    Next T
    Zu2=Random(Bedeut-1)+1
    Graphmode 1
    Deffill 0
    Pbox 240,146/Res,450,350/Res
    Deffill 1
    Bedeut=0
    Print At(18,9);"Vers.:";Zae
    Print At(18,11);"Gew.:";Gewusst
    Print At(58,11);"Oft.:";Vok(Zu)
    Print At(20,13);"Bedeutung : ";Vok$(Zu,Zu2)
    Print At(20,15);"Vokabel   : ";
    @Input(32,15,20,"")
    Bedeutung$=Upper$(B$)
    Flag=0
    If Bedeutung$=Upper$(Vok$(Zu,0))
          And Not Bedeutung$=""
      Flag=1
      Gflag=1
      Print At(32,17);"Richtig"
    Endif
    If Flag=0
      If Vok(Zu)<>0
        Dec Vok(Zu)
      Endif
      Print At(32,17);"Falsch"
      Be=Bedeut
    Else
      Inc Vok(Zu)
      If Vok(Zu)=Oft
        Inc Oftgew
      Endif
    Endif
    If Gflag=1
      Gflag=0
      Inc Gewusst
    Endif
    Repeat
    Until Inkey$=Chr$(13)
    Exit If Oftgew=>Vok
  Loop
Return
'
' E I N G E B E N
'
Procedure Eingeben
```

```
@Window(120,120,400,200,"Vokabeln eingeben",1)
Do
  Deffill 0
  Pbox 150,192/Res,450,300/Res
  Deffill 1
  Repeat
    Print At(20,9);Fre(0)
    Print At(20,11);"Vokabel : ";Space$(20)
    @Input(30,11,20,"")
    Vok$(Vok,0)=B$
    For T=1 To 5
      Print At(20,12+T);"Bedeutung #";
       Str$(T);" : ";Space$(20)
      @Input(35,12+T,20,"")
      Vok$(Vok,T)=B$
      If Len(Vok$(Vok,T))=0
        T=5
      Endif
    Next T
    @Allesok
  Until Flag=2
  Inc Vok
  Exit If Vok$(Vok-1,0)=""
Loop
Dec Vok
Sput Screen$
Return
'
' A N S E H E N
'
Procedure Ansehen
  @Pruef
  @Window(120,120,400,210,"Vokabeln",1)
  For T=0 To Vok-1
    For X=0 To 5
      If X>0
        Tab=100
      Else
        Tab=0
      Endif
      Text 130+Tab,300/Res,Vok$(T,X)
      Get 122,190/Res,400,320/Res,A$
      Put 122,(190-16)/Res,A$
    Next X
    Repeat
    Until Mousek<>0
    If Mousek=2
      T=Vok
    Endif
  Next T
Return
'
' E D I T I R E N
'
Procedure Editieren
  @Pruef
  @Window(120,120,400,210,"Vokabeln Editieren",1)
  Flag=0
  Vvok=Vok
  Print At(20,9);
```

```
" Bitte zu editierende Vokabel eingeben !"
Repeat
  @Input(25,11,20,"")
  E$=B$
Until E$<>""
For T=0 To Vok
  If Instr(E$,Vok$(T,0))>0
    Evok=T
    Aflag=1
  Endif
Next T
If Aflag=1
  Repeat
    Print At(20,12);"Vokabel : ";Space$(20)
    @Input(30,13,20,Vok$(Evok,0))
    Vok$(Evok,0)=B$
    For T=1 To 5
      Print At(20,14+T);"Bedeutung #";
        Str$(T);" : ";Space$(20)
      @Input(35,14+T,20,Vok$(Evok,T))
      Vok$(Evok,T)=B$
      If Len(Vok$(Evok,T))=0
        For L=T To 5
          Vok$(Evok,L)=""
        Next L
        T=5
      Endif
    Next T
    @Allesok
  Until Flag=2
Else
  Alert 1,"Kann diese Vokabel nicht finden
    >>"+E$+"<<",1," OK ",Dummy
Endif
Aflag=0
Return
'
' S U C H E N
'
Procedure Suchen
  @Pruef
  @Window(120,120,400,250,"Vokabeln Suchen",1)
  Print At(20,11);"    Bitte zu suchende
    Vokabel eingeben"
  Do
    Print At(20,13);"Vokabel :                    "
    Repeat
      @Input(30,13,20,"")
    Until B$<>""
    Su$=B$
    For Z=0 To Vok
      For Tt=0 To 5
        If Upper$(Left$(Vok$(Z,Tt),
          Len(Su$)))=Upper$(Su$)
          Zei=0
          For T=0 To 5
            Print At(32,15+Zei);Vok$(Z,T)
             +Space$(20-Len(Vok$(Z,T)))
            Inc Zei
            If T=0
```

```
            Inc Zei
          Endif
        Next T
        Print At(35,15+Zei);"> RETURN <"
        Repeat
        Until Inkey$=Chr$(13)
        Repeat
        Until Inkey$=""
        Print At(35,15+Zei);"                "
      Endif
    Next Tt
  Next Z
Loop
Return
'
' A B S P E I C H E R N
'
Procedure Save_all
  @Pruef
  File$="\*.VOK"
  Fileselect File$,"",File$
  If File$<>""
    Defmouse 2
    Open "O",#1,File$
    For T=0 To Vok
      Text 13,20,Str$(Vok-T)
      For A=0 To 5
        Print #1,Vok$(T,A)
      Next A
    Next T
    Close #1
    Defmouse 0
  Endif
Return
'
' L A D E N
'
Procedure Load_all
  Graphmode 1
  T=-1
  File$="\*.VOK"
  Fileselect File$,"",File$
  If Exist(File$)=-1
    Defmouse 2
    Open "I",#1,File$
    Repeat
      Inc T
      Text 13,20/Res,Str$(T)
      For A=0 To 5
        Line Input #1,Vok$(T,A)
      Next A
    Until Eof(#1)<>0
    Vok=T
    Close #1
    Defmouse 0
  Endif
Return
'
' D R U C K E N
'
```

```
Procedure Drucken
  Alert 1,"Wirklich alle ¦ Vokabeln ausdrucken ",
   1," NEIN ¦  JA  ",A1
  If A1=2
    X%=Gemdos(17)     !
   Wenn kein Drucker angeschlossen bzw OFF LINE X%=0
    If X%=0
      Alert 1," KEIN Drucker angeschlossen ¦
       bzw. OFF LINE ",1,"  OK  ",A1
      @Main
    Endif
    Defmouse 2
    Lprint Chr$(27)+"M";Chr$(27)+"R"+Chr$(2);
    For T=0 To Vok
      Lprint Vok$(T,0)
      For X=1 To 5
        If Len(Vok$(T,X))>1
          Lprint Space$((25);Vok$(T,X)
        Endif
      Next X
      If Mousek=2
        T=Vok
      Endif
    Next T
    Defmouse 0
  Endif
Return
'
' P A R A M E T E R
'
Procedure Parameter
  @Window(120,120,400,200,"Prameter",1)
  Print At(18,11);
    "Wie oft muß eine Vokabel gewusst werden : "
  Repeat
    @Input(60,11,2,Str$(Oft))
  Until Val(B$)>0
  Oft=Val(B$)
Return
'
' B I L D S C H I R M A U F B A U
'
Procedure Screen
  If Xbios(4)=1
    Res=2
  Else
    Res=1
  Endif
  Restore
  Pbox 0,0,639,399/Res
  Deftext 1,0,0,4
  Deffill 1,2,4
  Prbox 0,0,639,399/Res
  Deffill 1,0,0
  For T=2 To 7
    Pbox 10,(T*45)/Res,80,(T*45+40)/Res
    Read A$
    Text 15,(T*45+22)/Res,A$
    Pbox 560,(T*45)/Res,630,(T*45+40)/Res
    Read A$
```

```
    Text 565,(T*45+22)/Res,A$
  Next T
  Pbox 560,4/Res,630,24/Res .
  Pbox 10,4/Res,80,24/Res
  Pbox 10,40/Res,630,80/Res
  Deftext 1,0,0,13/Res
  Text 163,65/Res,"Vokabel-Champ
      (c) 1987 by Jan Kubuschok"
  Deffill 1,1,1
  Graphmode 3
  Sget Screen$
Return
'
' M E N U E A B F R A G E
'
Procedure Watch_menue
  Mouse X,Y,K
  If K=1
    If (X>10 And X<80) Or (X>560 And X<630)
      For T=2 To 7
        If Y>(T*45)/Res And Y<(T*45+40)/Res
          If X>560
            Pbox 560,(T*45)/Res,630,(T*45+40)/Res
            @Mkey(0)
            Pbox 560,(T*45)/Res,630,(T*45+40)/Res
            Selected=T+6
          Else
            Pbox 10,(T*45)/Res,80,(T*45+40)/Res
            @Mkey(0)
            Pbox 10,(T*45)/Res,80,(T*45+40)/Res
            Selected=T
          Endif
        Endif
      Next T
    Endif
  Endif
Return
'
' U H R Z E I T & V O K A B E L  A N Z A H L
'
Procedure Clock
  If Time$<>Ti$
    Deftext 1,0,0,13/Res
    Graphmode 1
    Text 563,20/Res,Time$
    Text 13,20/Res,Vok
    Ti$=Time$
    Graphmode 3
  Endif
Return
'
' W A R T E N  B I S  M O U S E K  =  F L A G
'
Procedure Mkey(Flag)
  Repeat
  Until Mousek=Flag
Return
'
' W I N D O W
'
```

```
' x=X-Koordinate
' y=Y-Koordinate
' h=Höhe
' b=Breite
' win$=Text
' flag=Springbox ein oder aus (0,1)
'
Procedure Window(X,Y,H,B,Win$,Flag)
  Y=Y/Res
  B=B/Res
  If Flag=1
    Dpoke Gintin,640
    Dpoke Gintin+2,400/Res
    Dpoke Gintin+4,50
    Dpoke Gintin+6,50/Res
    Dpoke Gintin+8,X
    Dpoke Gintin+10,Y
    Dpoke Gintin+12,H
    Dpoke Gintin+14,B
    Gemsys 73
  Endif
  Graphmode 1
  Pbox X,Y,H+X,B+Y
  Graphmode 3
  Pbox X,Y,H+X,B+Y
  Graphmode 1
  Box X+2,Y+2,H-2+X,B-2+Y
  Pbox X+5,Y+B,X+H,(Y+B)+5/Res
  Pbox X+H,Y+5,H+5+X,(B+Y)+5/Res
  Text X+(H-(Len(Win$)*16)),Y+(20/Res),Win$
Return
'
' A L L E S   O K
'
Procedure Allesok
  Graphmode 1
  Flag=0
  Get 198,348/Res,442,392/Res,Block$
  Deffill 1,0
  Pbox 198,348/Res,442,392/Res
  Text 237,376/Res,"  NEIN"
  Text 327,376/Res,"    JA"
  Box 230,360/Res,310,380/Res
  Box 320,360/Res,410,380/Res
  Box 320-1,360-1/Res,410+1,380+1/Res
  Box 200,350/Res,440,390/Res
  Deffill 1,1,1
  Graphmode 3
  Do
    Mouse X,Y,K
    If X>230 And Y>360/Res And X<310
        And Y<380/Res And K=1
      Pbox 230,360/Res,310,380/Res
      @Mkey(0)
      Pbox 230,360/Res,310,380/Res
      Flag=1
    Endif
    If (X>320 And Y>360/Res And X<410
        And Y<380/Res And K=1) Or Inkey$=Chr$(13)
      Pbox 320,360/Res,410,380/Res
```

```
      @Mkey(0)
      Pbox 320,360/Res,410,380/Res
      Flag=2
    Endif
    Exit If Flag=1 Or Flag=2
  Loop
  Graphmode 1
  Put 198,348/Res,Block$
Return
'
' D A T A S
'
Data " Abfragen","  Laden"," Eingeben",
     Speichern," Ansehen",Ausdrucken,
     "Editieren","Statistik","  Suchen",
     "Parameter"," E X I T",".Löschen
'
' L O E S C H E N
'
Procedure Losch
  Alert 1,"Wirklich alle | Vokabeln löschen",
        1," NEIN |  JA  ",A1
  If A1=2
    Run
  Endif
Return
'
' P R U E F
'
Procedure Pruef
  If Vok=0
    Alert 3," STOP: |   |Keine Vokabeln im
                  Speicher",1," Na klar ",Dummy
    @Main
  Endif
Return
'
' S T A T I S T I K
'
Procedure Statistik
  @Pruef
  Gew1=0
  Gew2=0
  Gew3=0
  If Zae=0
    Alert 1," Es wurde noch nicht ausgefragt,
      |somit kann noch keine |Statistik erstellt werden
      ",1,"  OK  ",A1
    @Main
  Endif
  @Window(120,120,400,210,"Statistik",1)
  For T=0 To Vok
    If Vok(TT)=1
      Inc Gew1
    Endif
  Next T
  For T=0 To Vok
    If Vok(T)=2
      Inc Gew2
    Endif
```

```
    Next T
    For T=0 To Vok
      If Vok(T)=3
        Inc Gew3
      Endif
    Next T
    Gew1=(Gew1/Vok)*3600
    Gew2=(Gew2/Vok)*3600
    Gew3=(Gew3/Vok)*3600
    Deffill 1,2,5
    Pcircle 320,200/Res,50,0,Gew1
    Pcircle 200,200/Res,50,0,Gew2
    Pcircle 440,200/Res,50,0,Gew3
    Pbox 150,300/Res,180,310/Res
    Deffill 1,2,7
    Pcircle 320,200/Res,50,Gew1,3600
    Pcircle 200,200/Res,50,Gew2,3600
    Pcircle 440,200/Res,50,Gew3,3600
    Text 270,270/Res,"1x Gewußt "+Str$(Int(Gew1/36))+"%"
    Text 150,270/Res,"2x Gewußt "+Str$(Int(Gew2/36))+"%"
    Text 390,270/Res,"3x Gewußt "+Str$(Int(Gew3/36))+"%"
    Text 190,310/Res,"Gewußte Vokabeln"
    Deffill 1,1,1
    @Mkey(1)
Return
'
'   EIGENE INPUT ROUTINE
'   AUFRUFEN MIT : @INPUT(ZEILE,SPALTE,
'   MAX.LAENGE,STRING$)
'   B$ enthält dann den eingegebenen String
'
'
Procedure Input(Spalte%,Zeile%,Max,In$)
  Hidem
  Void Xbios(21,2)
  Print At(Spalte%,Zeile%);Chr$(27)+"e";In$;
  Print At(Spalte%,Zeile%);
  B$=""
  B$=In$+Space$(Max-Len(In$))
  Zaehler%=0
  Do
    @Clock
    A$=Inkey$
    A=Asc(A$)
    B=Asc(Mid$(A$,2,1))
    If (A>31 And A<160 And Zaehler%<Max)
    And Not A=127
      Inc Zaehler%
      Mid$(B$,Zaehler%,1)=A$
      Print At(Spalte%+Zaehler%-1,Zeile%);A$;
    Endif
    If A=27
      B$=Space$(27)
      Print At(Spalte%,Zeile%);Left$(B$,Max);
      Print At(Spalte%,Zeile%);
      Zaehler%=0
    Endif
    If A=8 And Zaehler%>=1
      Dec Zaehler%
      Print Chr$(27)+"D";
```

```
      Mid$(B$,Zaehler%+1,1)=" "
      Print Chr$(27)+"j";
      Print At(Spalte%,Zeile%);Left$(B$,Max);
      Print Chr$(27)+"k";
    Endif
    If B=82
      B$=Mid$(B$,1,Zaehler%)+" "
      +Mid$(B$,Zaehler%+1,Max-1)
      Print Chr$(27)+"j";
      Print At(Spalte%,Zeile%);Left$(B$,Max);
      Print Chr$(27)+"k";
    Endif
    If B=77 And Zaehler%<Max
      Print Chr$(27)+"C";
      Inc Zaehler%
    Endif
    If B=75 And Zaehler%=>1
      Print Chr$(27)+"D";
      Dec Zaehler%
    Endif
    Exit If A=13 Or B=97
  Loop
  Print Chr$(27)+"f";
  Showm
  If B<>97
    Repeat
      Dec Max
    Until Mid$(B$,Max,1)<>" " Or Max=0
    B$=Left$(B$,Max)
  Else
    @Main
  Endif
Return
'
' E X I T
'
Procedure Exit
  Alert 1,"Wircklich abbrechen ??",1," NEIN |  JA ",A1
  If A1=2
    Edit
  Endif
Return
'
' F E H L E R
'
Procedure Fehler
  If Err=16
    Alert 1," Kann keine Vokabeln mehr |
      aufnehmen. | Bis zu 1000 Vokabeln
        | möglich.",1," OK ",Dummy
    Close #1
    @Main
  Endif
Return
```

Problemlose Tastatur Umbelegung

Wollten Sie nicht schon immer die Tastatur Ihres ST's umbelegen, um z.B. den 10er Block als Hex-Tastatur benutzen zu können?

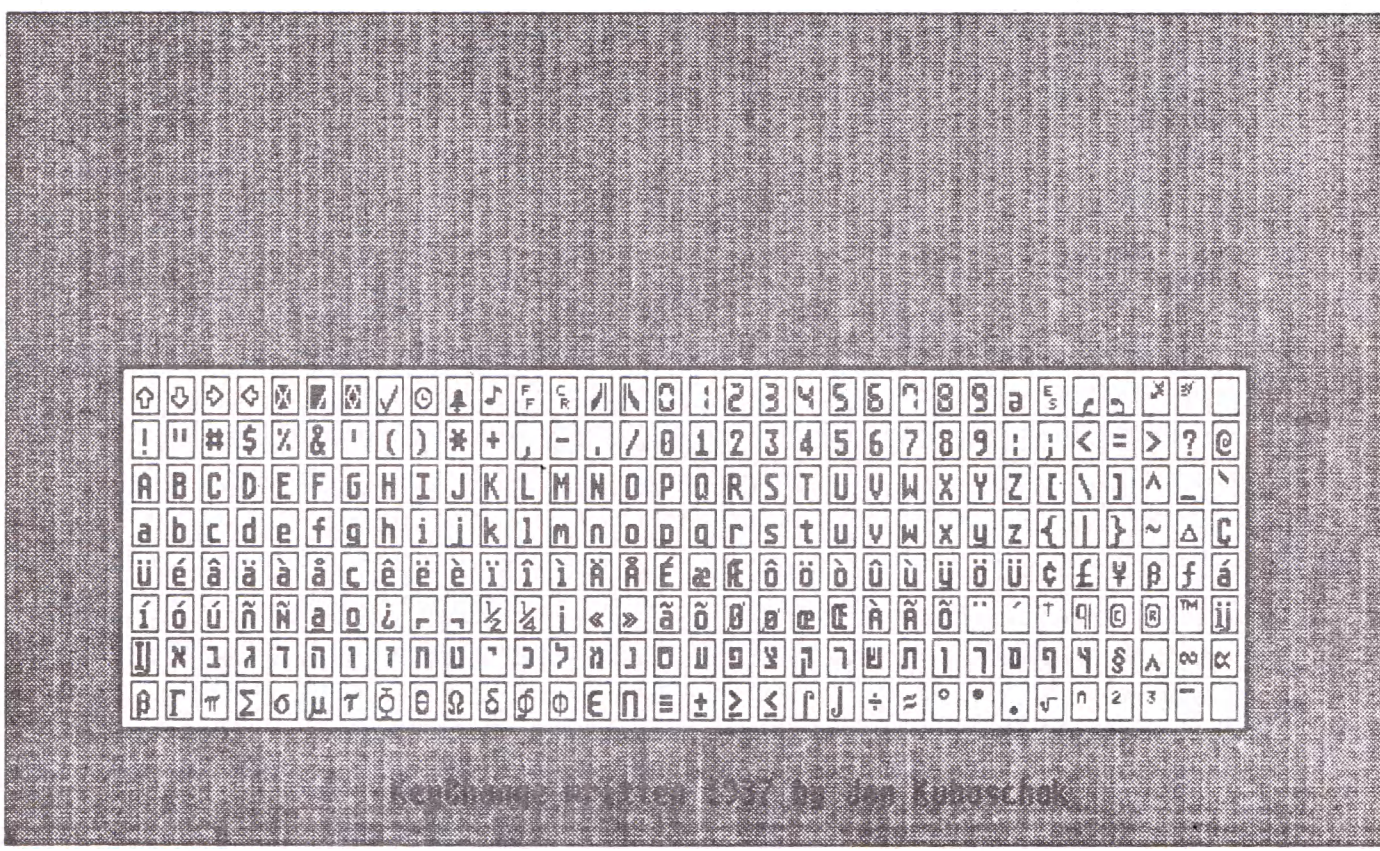
Dieses Programm gibt Ihnen nun die Möglichkeit, Ihre Tastatur so zu belegen, wie Sie es wünschen. Mit dem kleinen Hilfsprogramm können Sie dann, nachdem Sie das Programm compiliert haben, im Auto-Ordner die geänderte Tastaturbelegung einladen. Die geänderte Belegung wird z.B. von 1ST WORD oder anderen Textverarbeitungs-Programmen genutzt.
Die Belegung wird immer dann zerstört, wenn ein XBIOS(24) aufgerufen wird oder der Bereich oberhalb des VIDEO-RAMS (> Xbios(2)+ 3128) überschrieben wird. Es empfiehlt sich auch ein Programm, in dem nur ein xbios(24) aufgerufen wird, anzulegen, da mit der Tastatur nicht mehr viel zu machen ist, wenn die Umbelegungstabelle zuerstört wird. Nach Xbios(24) ist wieder die ganz normale ST .Tastaturbelegung, wie Sie sie schon immer gewohnt waren, vorhanden.

Nun noch etwas näher zu den beiden Programmen:

Mit dem ersten Programm (KEY-CHANGE), eigentlich das Hauptprogramm, können Sie Ihre Tastatur beliebig abändern. Drücken Sie zuerst die entsprechende Taste, um den Scan-Code zu erhalten. Dabei kann auch gleichzeitig SHIFT oder CAPSLOCK gedrückt bzw. aktiviert sein. Nun können Sie aus dem Rechteck mit dem Zeichensatz das gewünschte Zeichen auswählen. Während dieser Auswahl ist auch das Pull-Down-Menue aktiviert. Sie können nun die Belegung abspeichern oder laden, die Belegung wieder rückgängig machen und schließlich auch das Programm verlassen.

Mit dem zweiten Programm (BLOAD), können Sie eine erstellte Tastaturtabelle laden, dabei muß die zu ladende Tabelle BELEGUNG. KEY heißen. Dieses Programm arbeitet im 640x400 Pixel Modus. Leute mit Farbmonitor können sich ja bei einem Bekannten eine Tabelle erstellen und diese dann mit dem Installationsprogramm (unabhängig von der Auflösung) auf ihrem ST installieren.

jk



Die aktuelle Belegung der ST-Tastatur

```
'
' KEY-CHANGE
'
'
@Screen
'
' Pull-Down Menue
'
Dim Eintrag$(20)
For I%=0 To 20
  Read Eintrag$(I%)
Next I%
Menu Eintrag$()
On Menu  Gosub Action
'
'
On Break Gosub Fine
'
' CapsLock Taste ausschalten
'
Void Bios(11,Bios(11,-1) And &HEF)
'
' ASCII-Feld aufbauen
'
@Show
'
' Anfangsadresse der Belegungstabelle holen
' und Buffer vorbereiten
'
Pointer%=Xbios(16,L:-1,L:-1,L:-1)
Dim Buf%(96)
Buf%=Varptr(Buf%(0))
Bmove Lpeek(Pointer%),Buf%,128
Bmove Lpeek(Pointer%+4),Buf%+128,128
Bmove Lpeek(Pointer%+8),Buf%+256,128
'
' MAIN
'
Do
  Deftext 1,2,0,13
  Text 24,14,"DESK  BELEGUNG"
  Graphmode 1
  Put 55,183,Zeichen$
  Deftext 1,0,0,4
  Text 400,8,"Scancode  :          "
  Scancode=0
  @Get_scan_code
  Graphmode 1
  Text 470,8,Str$(Scancode)+"  "
  Text 400,16,"ASCII-Code:          "
  Asciicode=0
  Deftext 1,0,0,13
  Text 24,14,"DESK  BELEGUNG"
  @Get_ascii_code
  Graphmode 1
  Deftext 1,0,0,4
  Text 470,16,Str$(Asciicode)+"  "
  Poke Buf%+Scancode,Asciicode
  Void Xbios(16,L:Buf%,L:Buf%+128,L:Buf%+256)
Loop
```

```
'
' Unterprogramme
'
' Zeigt Belegung
'
Procedure Zeige
  Deftext 1,0,0,6
  Sget Screen$
  Cls
  X=0
  Y=0
  For T=0 To 127
    Inc Y
    Text X*200,Y*8,"  Scancode :"+Str$(T)+
      "  ASCII:"+Str$(Peek(Buf%+T))
    If T=40 Or T=81
      Inc X
      Y=0
    Endif
  Next T
  Print At(15,23);"Ohne Shift        >Taste<"
  Void Inp(2)
  Cls
  X=0
  Y=0
  For T=128 To 256
    Inc Y
    Text X*200,Y*8,"  Scancode :"+Str$(T)+
     "  ASCII:"+Str$(Peek(Buf%+T))
    If T=40+128 Or T=81+128
      Inc X
      Y=0
    Endif
  Next T
  Print At(15,23);"mit Shift         >Taste<"
  Void Inp(2)
  X=0
  Y=0
  Cls
  For T=256 To 256+128
    Inc Y
    Text X*200,Y*8,"  Scancode :"+Str$(T)+
     "  ASCII:"+Str$(Peek(Buf%+T))
    If T=40+256 Or T=81+256
      Inc X
      Y=0
    Endif
  Next T
  Print At(15,23);"CapsLock          >Taste<"
  Void Inp(2)
  Sput Screen$
Return
'
' Zeigt ASCII Feld
'
Procedure Show
  Deftext 1,0,0,13
  Graphmode 1
  A=0
  Deffill 1,0,0
```

```
  Pbox 52,180,571,347
  Box 53,181,570,346
  For Y=0 To 7
    For X=0 To 31
      Inc A
      Text X*16+60,Y*20+200,Chr$(A)
      Box X*16+60-2,Y*20+202,X*16+70,Y*20+185
    Next X
  Next Y
  Get 55,183,568,344,Zeichen$
Return
'
' Schaltet Rahmen bei Box
' Circle,Ellipse usw. aus (0)
' bzw. ein  (1)
'
Procedure Umrandung(Flag)
  Dpoke Contrl+2,0
  Dpoke Contrl+6,1
  Dpoke Contrl+12,2
  Dpoke Intin,Flag
  Vdisys 104
Return
'
' Holt den Scancode
'
Procedure Get_scan_code
  On Menu
  A=Bios(2,2)   ! Scancode
  Status=Bios(11,-1) And (2 Or 16)
  If Status=2
    Status=1
  Endif
  If Status=16
    Status=2
  Endif
  If Status<>1 And Status<>2
    Status=0
  Endif
  Scancode=A/65535 And 255
  Scancode=Scancode+(128*Status)
Return
'
' Holt ASCII Code
'
Procedure Get_ascii_code
  Graphmode 3
  Flag=0
  Put 55,183,Zeichen$
  Do
    On Menu
    Mouse Mx,My,Mk
    If Mk=1
      For Y%=0 To 7
        For X%=0 To 31
          If Mx>X%*16+60-2 And My<Y%*20+202
           And Mx<X%*16+70 And My>Y%*20+185
            Put 55,183,Zeichen$
            Asciicode=Y%*32+X%+1
            Pbox X%*16+60-2,Y%*20+202,X%*16+70,Y%*20+185
```

```
            Flag=1
          Endif
          Exit If Flag=1
        Next X%
        Exit If Flag=1
      Next Y%
    Endif
    Exit If Flag=1 Or K=2
  Loop
Return
'
' Programmende
'
Procedure Fine
  Alert 1,"restore ??",1,"  Ja  | Nein ",A1
  If A1=1
    Void Xbios(24)
  Endif
  Edit
Return
'
' Bildschirmaufbau
'
Procedure Screen
  Deftext 1,0,0,13
  @Umrandung(1)
  Deffill 1,1,1
  Pbox 0,0,639,399
  Deffill 1,2,4
  Prbox 0,0,639,399
  Graphmode 2
  Text 170,380,"
Return
'
' Pull-Down-Menue ??
'
Procedure Action
  If Menu(0)=14
    Void Xbios(24)
  Endif
  If Menu(0)=16
    Gosub Zeige
  Endif
  If Menu(0)=18
    Edit
  Endif
  If Menu(0)=12
    Gosub Laden
  Endif
  If Menu(0)=11
    Gosub Speichern
  Endif
  If Menu(0)=1
    Gosub Info
  Endif
  '
  If Menu(0)=22      ! MENÜPUNKT 25
```

```
    Menu Kill
    End

    Endif
    Menu Off
  Return
  '
  ' Laden
  '
  Procedure Laden
    File$="\*.*"
    Fileselect File$,"",File$
    If Exist(File$)=-1
      Open "I",#1,File$
      For T=0 To 96
        Input #1,Buf%(T)
      Next T
      Close #1
      Void Xbios(16,L:Buf%,L:Buf%+128,L:Buf%+256)
    Endif
  Return
  '
  ' Speichern
  '
  Procedure Speichern
    File$="\*.*"
    Fileselect File$,"",File$
    If File$<>""
      Open "O",#1,File$
      For T=0 To 96
        Print #1,Buf%(T)
      Next T
      Close #1
    Endif
  Return
  Procedure Info
    Alert 0,"   (C) 1987 by |
      | Jan Kubuschok ",1," OK ",A1
  Return
  '
  ' Datas für Pull-Down-Menue
  '
  Data DESK,  KeyChange
  Data ----------------
  Data 1,2,3,4,5,6,""
  Data BELEGUNG, Speichern ,   Laden  ,----------
  Data   RESTORE ,----------,  Zeigen  ,----------
  Data   ENDE ,""
  Data "",""
```

```
'
' BLOAD by Jan Kubuschok
'

Start%=Xbios(2)+32128
File$="belegung.key"
Print Chr$(27)+"p"
Print At(1,1);"
     KeyChange v1.0
Print Chr$(27)+"q"
Void Bios(11,Bios(11,-1) And &HEF)
Pointer%=Xbios(16,L:-1,L:-1,L:-1)
Dim Buf%(96)
Buf%=Varptr(Buf%(0))
Bmove Lpeek(Pointer%),Buf%,128
Bmove Lpeek(Pointer%+4),Buf%+128,128
Bmove Lpeek(Pointer%+8),Buf%+256,128
If Exist(File$)=-1
  Open "I",#1,File$
  For T=0 To 96
    Input #1,Buf%(T)
  Next T
  Close #1
  Bmove Buf%,Start%,384
  Void Xbios(16,L:Start%,L:Start%+128,L:Start%+256)
  Print "Installiert !"
  Pause 10
Else
  Print
  Print " Kann File nicht finden"
  Void Xbios(24)
  End
Endif
Exec 0,File$,"",""
```

Tips & Tricks

```
' Mit PSAVE gespeicherte
' Programm entschützen
Do
  Alert 1," PSAVE Flag setzen oder
    | löschen ?",3,"lösche|setze|ABBRUCH",A1
  If A1
    Fileselect "\*.BAS","",File$
    If Len(File$)>0
      Open "U",#1,File$
      If A1=1
        Out #1,0
      Endif
      If A1=2
        Out #1,&HFF
      Endif
      Close #1
    Endif
  Endif
  Exit If A1=3
Loop
```

ST am Farb-fernseh-gerät

Nicht jeder hat das Geld, sich einen Farbmonitor zu kaufen, andererseits aber sind manche Games nur in Farbe überhaupt interessant. Unser Tip: Passen Sie Ihr Farb-TV an den Atari an!

Der Atari ST verfügt über drei verschiedene Bildschirmauflösungen. In der höchsten Auflösungsstufe, in der 640 * 400 Punkte in zwei Farben dargestellt werden, arbeiten die meisten kommerziellen Programme (z.B. Textverarbeitung), da der für diesen Modus vorgesehene Monitor SM124 ein hervorragendes monochromes Bild liefert, das ein ermüdungsloses Arbeiten am Computer gewährleistet.

Die meisten Grafikprogramme und fast alle Spiele verwenden hingegen die beiden Farbmodi, in denen 4 bzw. 16 Farben dargestellt werden können. Diese Bildschirmauflösungen kann man mit dem Schwarz-Weiß-Monitor nicht nutzen. Wenn man alle Darstellungsstufen benutzen wollte, müßte man sich also zwei Monitore zulegen, die zusammen rund 1500 DM kosten würden. Die große Mehrheit aller ST-Besitzer hat sich für den monochromen Monitor entschieden und ist nicht bereit, etwa 1000 DM für den Farbmonitor auszugeben.

Es gibt jedoch eine recht preiswerte Möglichkeit, den heimischen Farbfernseher an den ST anzuschließen. Fast alle relativ neuen Geräte verfügen über einen sogenannten Scart-Anschluß. Dieser wird oft auch als „Euro-AV-Buchse" bezeichnet und besitzt 21 Pole. Der Scart-Eingang eines Fernsehers wird unter anderem zum Anschluß von Videorekordern, HiFi-Anlagen und – wie in unserem Fall – Computern benutzt.

Um einen Atari ST mit Hilfe dieser Buchse an den Fernseher anzuschließen, benötigt man nur noch ein sogenanntes Scart-Kabel, das bei vielen Atari-Fachhändlern für weniger als 100 DM erhältlich ist.

Eine weitere Möglichkeit bieten HF-Modulatoren, die eine Verwendung eines beliebigen Fernsehers ermöglichen. Die Bildqualität ist im Vergleich zur Benutzung eines Scart-Eingangs jedoch weitaus schlechter.

Wenn Sie beim Einsatz eines Fernsehers trotz guter Lesbarkeit der Texte ein starkes Flimmern des Bildes feststellen, so liegt dies nicht am Fernsehgerät, sondern am Atari ST. Doch keine Sorge, Ihr Rechner ist nicht etwa defekt. Vielmehr liefert der Computer die Bildschirminformationen lediglich mit einer falschen Bildwechselfrequenz (50 Hz). Diese kann man jedoch mit Hilfe des hier abgedruckten kleinen Maschinenprogramms softwaremäßig auf 60 Hz erhöhen.

Das Programm schaltet die Bildwechselfrequenz von 50 Hz auf 60 Hz um, wenn es durch Anklicken des Icons gestartet wird. Die Bildschirmdarstellung sollte sich dann verbessern. Um wieder auf 50 Hz zurückzuschalten, muß das Programm nur erneut gestartet werden.

Die Maschinenroutine ist neben der Source-Code-Darstellung auch in Form von DATA-Zeilen als Basic-Programm abgedruckt, damit sie auch von Lesern, die nicht über einen Assembler verfügen, nutzbar ist.

Wenn man das Basic-Programm eingegeben hat, sollte man es vor dem Starten sicherheitshalber abspeichern. Nach RUN fragt das Programm nach dem Namen der zu erzeugenden Programmdatei. Anschließend werden die Data-Zeilen in ein lauffähiges Maschinenprogramm umgesetzt und in dieser Form auf der eingelegten Diskette abgespeichert. Der Computer überprüft dabei automatisch, ob die Datas fehlerfrei sind. Fehlerhafte Zeilen werden auf dem Bildschirm angezeigt. Die auf diese Weise erhaltene Programmdatei ist mit dem durch Assemblierunt des Source-Codes erzeugten File identisch.

Es bietet sich natürlich an, das Maschinenprogramm mit Hilfe eines Auto-Ordners beim Systemstart automatisch starten zu lassen, so daß stets eine gute Bildschirmdarstellung gewährleistet ist. Auf die Qualität des monochromen Monitorbildes hat die Umschaltung mit diesem Programm keinerlei Auswirkungen.

```
**********************************************
*                                            *
*                                            *
*                 ATARI ST                   *
*          Umschaltprogramm 50/60 Hz         *
*                   für                      *
*          Fernsehgeräte und Monitore        *
*                                            *
*                                            *
*                                            *
*                                            *
*                                            *
*                                            *
*                                            *
**********************************************
```

```
          TEXT

Adresse   equ $FFFF820A          enthält Frequenz
Supexec   equ $26                Supervisor-Mode
XBIOS     equ $E                 XBIOS-Trap
GEMDOS    equ $01                GEMDOS-Trap

Start     move.l #Umschalt,-(sp) Auf Supervisor-
          move.w #Supexec,-(sp)  Modus umschalten
          trap #XBIOS            und die Routine
          addq.l #6,sp           'Umschalt' starten

Ende      clr -(sp)              Programm beenden
          trap #GEMDOS

Umschalt  bchg #1,Adresse        Frequenz wechseln
          rts                    Routine beenden

          END
```

```
100   REM ************************************************************
110   REM *                                                          *
120   REM *          Umschaltprogramm 50/60 Hz                       *
130   REM *                für den ATARI ST                          *
140   REM *                 BASIC-LOADER                             *
150   REM *                                                          *
160   REM *                                                          *
170   REM *                                                          *
180   REM ************************************************************
190   REM
200   FULLW 2 : CLEARW 2
210   GOTOXY 15,2
220   PRINT "Frequenzumschaltung"
230   GOTOXY 2,5
240   INPUT "Filename für die Programmdatei : ",F$
250   GOTOXY 2,7
260   INPUT "Bitte Diskette einlegen und RETURN drücken ! ",IP$
270   GOTOXY 15,10 : PRINT "Bitte warten"
280   REM
290   REM ------ DATAs einlesen -------
300   REM
310   ZEILE = 600 : REM 600 ist erste DATA-Zeile
320   READ LAENGE
330   DIM F%(LAENGE/2+10)
340   START = VARPTR(F%(0))
350   FOR I=0 TO LAENGE/2 STEP 7
360   SUMME%= 0 : REM Checksumme löschen
370   ZEILE = ZEILE + 10
380   FOR J=I TO I+6
390   READ WERT%
400   SUMME%= SUMME%+ WERT%
410   F%(J)= WERT%
420   NEXT J
430   READ CHECK%
440   IF CHECK%<> SUMME%THEN 500 : REM Fehler in den DATAs
450   NEXT I
460   GOTOXY 12,15
470   PRINT "Keine Fehler in den DATAs"
480   BSAVE F$,START,LAENGE
490   END
500   REM ----- Fehler -----
510   GOTOXY 12,15
520   PRINT "Fehler in Zeile";ZEILE;"!!!"
530   GOTOXY 12,17
540   INPUT "Bitte RETURN drücken !!!",IP$
550   END
560   REM
570   REM ------- DATA-Zeilen -------
580   REM
600   DATA 61
610   DATA  24602, 0, 28, 0, 0, 0, 0, 24630
620   DATA  0, 0, 0, 0, 0, 0, 0, 0
630   DATA  12092, 0, 18, 16188, 38, 20046, 23695, 6541
640   DATA  16999, 20033, 2169, 1,-1,-32246, 20085, 27040
650   DATA  0, 2, 65, 2169, 1,-1,-32246,-30010
```

Der Bildschirmschoner: Schaltet den Monitor aus

Das Bild auf einer Bildröhre eines Monitors oder Fernsehers entsteht durch die sogenannte Fluoreszenzschicht. Sie wird beim Auftreffen der Elektronen des Elektronenstrahls kurz zum Leuchten angeregt. Kann es nun passieren, daß einige Stellen immer das gleiche Muster oder Farben darstellen müssen, so besteht die Gefahr, daß sich das Bild in die Leuchtschicht einbrennt und die Qualität des Monitor stark herabsetzt.

In vielen Computern wird dieses Einbrennen durch Wechseln der Farben oder Abschalten des Bildes nach längeren Eingabepausen verhindert. Der ATARI ST ist leider nicht von Haus aus mit dieser Möglichkeit ausgestattet. Das läßt sich jedoch leicht ändern, wenn Sie das folgende Programm benutzen.

Die Funktion ist folgende: Wird eine bestimmte Zeit keine Mausbewegung oder Tastatureingabe vom Rechner festgestellt, so schaltet der ST einfach ein Monitorbild aus. Das Bild erscheint, sobald wieder eine Eingabe erfolgt ist und es kann sofort normal weitergearbeitet werden.

Wie funktioniert das Es wird die Zeit gemessen, in der keine Eingabe vom Benutzer vorgenommen worden ist. Ist eine bestimmte Zeit überschritten, dann stellt es den Shifterbaustein im ST auf externe Syncronisation um. Der Shifter ist zuständig für den Bildaufbau und kann entweder selber die Syncronisation übernehmen oder diese Aufgabe einem externen Gerät überlassen. Externe Synchronisation bedeutet aber im Normalfall kein Bild, also Sicherheit vor einem Einbrennen!
Bei Betätigung einer Taste oder Bewegung der Maus wird alles einfach wieder zurückgestellt und das Bild ist wieder da.

Das Listing wurde mit dem Ideal-Assembler erstellt. Es läßt sich natürlich leicht auch mit jedem anderen Assembler eingeben. Das fertige Programm ist gerade 200 Bytes lang und findet in jedem AUTO-Ordner Platz.

NOTFALLS DAS BASICLISTING ABTIPPEN

Sollte man über keinen Assembler verfügen, so kann das GFA-Basiclisting abgetippt werden. Es erzeugt schon das fertige Programm in Form eines Files auf Diskette.

```
**********************************
* Screen Saver -- save.s         *
*                                *
*                                *
*                                *
*                60              ; Anzahl der Sekunden
SYNCO           =      $FF820A   ; Hardwarereg. des Shifters
VBLQUEUE        =      $456      ; Pointer auf VBL-Queue
SEC             =      N*60      ; 1/60 sec.
**********************************
                .TEXT
START:
        MOVE.L  A7,A5           ;
        MOVE.L  4(A5),A5        ; Anzahl der Bytes
        MOVE.L  $C(A5),D7       ; des Prg's berechnen
        ADD.L   #$100,D7        ;

        PEA     INIT            ; Installation der Interruptrotinen
        MOVE    #38,-(SP)       ; natuerlich im Supervisormodus.
        TRAP    #14             ; (Super_exec)
        ADDQ.L  #6,SP           ;

        CLR     -(SP)           ; Programm verlassen
        MOVE.L  D7,-(SP)        ; Speicherbereich nicht
        MOVE    #$31,-(SP)      ; freigeben.
        TRAP    #1              ;

**********************************
INIT:                           ;
        CMP.L   #'JENS',$420    ; Bereits installiert?
        BNE.S   IN02            ; nein -->
        CLR.L   D7              ; Keinen Speicher retten!
        RTS                     ; Und zurueck...
IN02:   MOVE.L  $118,OLDKEY     ; Neue Interruptroutinen
        MOVE.L  #NEUKEY,$118    ; einfuegen.

        MOVE.L  VBLQUEUE,A0     ; VBL_Queue nach freiem
IN01:   TST.L   (A0)+           ; Eintrag durchsuchen.
        BNE.S   IN01
                                ; gefunden!
        CLR     COUNT           ; Zähler zurücksetzen.
        MOVE.L  #NEUVBL,-(A0)   ; Routine eintragen.
        MOVE.L  #'JENS',$420    ; Installation erfolgt.
        RTS                     ;
COUNT:  DS.W    1               ; Zaehler.
```

```
**********************************
NEUVBL:
        ADD     #1,COUNT        ; Zähler erhöhen.
        CMP     #SEC,COUNT      ; Schon abgelaufen ?
        BNE.S   NE01            ; nein -->
        OR.B    #1,SYNCO        ; Externe Syncronisation!
NE01:
        RTS
**********************************
NEUKEY:
        AND.B   #$FE,SYNCO      ; Syncronisation ein.
        CLR     COUNT           ; Zähler zurücksetzen.
        DC.B    $4E,$F9         ; JMP
OLDKEY: DC.L    -1              ; alte Routine ausführen.
```

GFA-Basiclisting

```
' ------------------------------------------------------'
' Dateigenerator                                        '
' ------------------------------------------------------'
Print "Datei mit Namen ";
Fileselect "\*.*",B$,Dat$
Print Dat$;" erzeugen."
Open "O",#1,Dat$
Zeile=0
Sum=0
Do
  Read A
  Exit If A=-1
  If A<10000
    Sum=Sum+A
    Out #1,A
  Else
    Flag=A-10000-Sum
    Inc Zeile
    Sum=0
  Endif
  Exit If Flag<>0
Loop
Close #1
If A=-1
  Print "Alle DataZeilen ok!"
Else
  Print "Datafehler in Data-Zeile No. ";Zeile
Endif
End
Data 96,26,0,0,0,150,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,10272
Data 0,0,0,42,79,42,109,0,4,46,45,0,12,6,135,0,0,1,0,72,121,10714
Data 0,0,0,40,63,60,0,38,78,78,92,143,66,103,47,7,63,60,0,49,10987
Data 78,65,12,184,74,69,78,83,4,32,102,4,66,135,78,117,35,248,11464
Data 1,24,0,0,0,146,33,252,0,0,0,130,1,24,32,120,4,86,74,152,11079
Data 102,252,66,121,0,0,0,100,33,60,0,0,0,102,33,252,74,69,11264
Data 78,83,4,32,78,117,0,0,6,121,0,1,0,0,0,100,12,121,14,16,10783
Data 0,0,0,100,102,8,0,57,0,1,0,255,130,10,78,117,2,57,0,254,11171
Data 0,255,130,10,66,121,0,0,0,100,78,249,255,255,255,255,0,12029
Data 0,0,18,40,6,16,6,20,8,26,10140,-1
```

Printer Spooler für den Atari ST

Sicherlich haben Sie sich schon darüber geärgert, daß Ihr Drucker den Computer während der Druckzeit blockiert und der Computer eigentlich überhaupt nicht ausgelastet ist.

Andere Rechner, die über ein Betriebssystem mit Multitasking verfügen, erledigen das Drucken sozusagen nebenher. Unser ST ist damit jedoch bekanntlich nicht ausgerüstet. Hier hilft unser Druckerspooler weiter. Er läßt den ST während des Ausdrucks auch noch andere Aufgaben erledigen.

WIE FUNKTIONIERT DER SPOOLER?

Aufgabe des Computers ist es, beim Ausdruck die Zeichen auf den Centronicsport zu geben. Danach wartet er auf das Busy-Signal, womit der Drucker ihm mitteilt, daß das Zeichen verarbeitet worden ist und er ein neues Zeichen senden darf.

Der Spooler schickt alle Druckzeichen nicht direkt an den Drucker, sondern speichert sie in einem Druckpuffer im RAM ab. Der Puffer wird dann entsprechend der Druckgeschwindigkeit wieder geleert und Zeichen für Zeichen auf den Centronicsport geschickt.

Das Senden der Zeichen geschieht mit Hilfe des Vertikal Blank Interrupts. Er erledigt periodisch alle 70 bzw. 50 Sekunden verschiedene Aufgaben im ST. Wir lassen ihn nun jedesmal prüfen, ob der Drucker ein Zeichen aus dem Puffer empfangen kann und senden es gegebenenfalls.

Der Puffer kann variabel je nach Größe des freien Speicherplatzes definiert werden. Das Programm ist so gehalten, daß es möglichst wenig Speicherplatz verbraucht. Es wurde vollständig in Assembler geschrieben und sollte eigentlich mit allen anderen Anwendungen zusammenarbeiten.

Für Leute, die keinen Assembler zur Verfügung haben, ist ein Basic-Listing abgedruckt.

ASSEMBLERLISTING

```
****************************************

* SPOOL.S                              *

****************************************

SAVPTR ..... =  $4A2
VBLQUEUE ..... =  $456
XBIOS ..... =  14
GEMDOS ..... =  1
SUPEXC ..... =  38
KEEP ..... =  $31
PRT ..... =  $9
READ ..... =  $A
BASPAG ..... =  4
HITPA ..... =  4

****************************************

START:

BRA     INSTALL

****************************************

TRAPNEU:

LEA     6(SP),A0        ; Funktionsnummer
MOVE    (SP),D0         ; SR-Register
BTST    #13,D0          ; Supervisoraufruf ?
BNE     super           ; ja.
MOVE.L  USP,A0          ; Sonst USP
                          verwenden.

SUPER:

MOVE    0(A0),D0        ; Funktionsnummer
                          holen.
CMP     #3,D0           ; 'CONOUT' ?
BNE     DOOLD           ; nein.
MOVE    2(A0),D0        ; Centronics ?
BEQ     DOIT            ; ja.

DOOLD:

MOVE.L  TRAPOLD,A0      ; Alte Trapoutine
                          ausführen.
JMP     (A0)

****************************************

DOIT:

MOVE.L  SAVPTR,A1       ; Zeiger auf Reg. area
MOVE    (SP)+,D0        ; SR retten
MOVE    D0,-(A1)
MOVE.L  (SP)+,-(A1)     ; PC retten
MOVEM.L D3-D7/A3-A7,
        -(A1)           ; Register retten
MOVE.L  A1,SAVPTR       ; savptr update

WAFU:

MOVE.L  COUNT,D0        ; Puffer schon voll ?
CMP.L   LENGTH,D0
BGT     WAFU            ; Ja, warte . . .
MOVE.B  5(A0),D1        ; Character nach D1.
LEA     PUFFER,A1       ; Pufferstart.
MOVE.L  INP,D2          ; Pufferoffset IN.
MOVE.B  D1,0(A1,D2.L)   ; Zeichen abspeichern.
ADDQ.L  #1,COUNT        ; Zeichenzähler
                          erhöhen.
ADDQ.L  #1,D2           ; Pufferoffset erhöhen.
CMP.L   LENGTH,D2       ; Pufferende erreicht ?
BLE     MONI            ; nein.
CLR.L   D2              ; Offset zurücksetzen.

MONI:

MOVE.L  D2,INP          ; Neuen Zeiger
                          abspeichern.
MOVEQ.L #-1,D0          ; Keinen Fehlercode.
```

```
MOVE.L   SAVPTR,A1      ; Register restaurieren.
MOVEM.L  (A1)+,D3–D7/
         A3–A7
MOVE.L   (A1)+,–(SP)
MOVE     (A1)+,–(SP)
MOVE.L   A1,SAVPTR

RTE

****************************************

VBLROUT:

TST.L    COUNT          ; Puffer leer ?
BEQ      VBLEND         ; ja.
BTST     #0,$FFFFFA01   ; Busy testen.
BNE      VBLEND         ; busy . . .
MOVE     SR,D6          ; Status retten.
OR       #$700,SR       ; Interrupts sperren.
LEA      $FFFF8800,A1   ; Soundchip Register.
MOVE.B   #7,(A1)        ; Mixer selektieren.
MOVE.B   (A1),D0        ; Mixer lesen.
OR.B     #$80,D0        ; Port B auf Ausgabe.
MOVE.B   D0,2(A1)       ; Mixer schreiben.
MOVE.B   #15,(A1)       ; Port B selektieren.
MOVE.L   OUTP,D1        ; Offset für Output
                          holen.
LEA      PUFFER,A0
MOVE.B   0(A0,D1.L),D0  ; Zeichen aus Puffer
                          holen.
MOVE.B   D0,2(A1)       ; Port B schreiben
MOVE.B   #14,(A1)       ; Port A selektieren.
MOVE.B   (A1),D0        ; Port A lesen.
OR.B     #$20,D0        ; Strobe Low.
MOVE.B   D0,2(A1)
AND.B    #$DF,D0        ; Strobe High.
MOVE.B   D0,2(A1)
MOVE     D6,SR          ; Statusreg. zurück-
                          holen.
SUBQ.L   #1,COUNT       ; –1 Zeichen.
ADDQ.L   #1,D1
CMP.L    LENGTH,D1
BLE.S    VBLO
CLR.L    D1

VBLO:

MOVE.L   D1,OUTP

VBLEND:

TST.L    VBLOLD
BEQ      VBLRET
MOVE.L   VBLOLD,A0
JMP      (A0)

VBLRET:

RTS

****************************************

VBLOLD:

DC.L     0              ; Alter VBL-Vektor

TRAPOLD:

DC.L     0              ; Alter Trapvector
                          #13

LENGTH:

DC.L     0              ; Pufferlänge.

COUNT:

DC.L     0              ; Anzahl der Zeichen
                          im Puffer.

INP:

DC.L     0              ; Pointer IN-Puffer.

OUTP:

DC.L     0              ; Pointer OUT-Puffer.

PUFFER:

                        ; Hier beginnt der
                          Druckerpuffer.

****************************************

INSTALL:

MOVE.L   SP,A5          ; Basepage holen
MOVE.L   BASPAG(A5),A5
MOVE.L   HITPA(A5),D7   ; TPA Ende
SUB.L    #INSTALL,D7    ; Freien Platz
                          berechnen.
DIVU     #1024,D7       ; in KByte.
SUB      #128,D7        ; Minus Sicherheit.
MOVE     D7,D0          ; D7 = freier Speicher.
EXT.L    D0
LEA      ZIFF,A0        ; Ziffer in Text
                          einfügen.
BSR      WANDEL         ;
LEA      STRING1,A0
BSR      PRINT
PEA      INBUFF         ; 'Readline'
MOVE     #READ,–(SP)
TRAP     #GEMDOS
LEA      6(SP),SP
LEA      INBUFF+1,A0    ; KByte eingeben
CLR.L    D2             ; In Binaer umrechnen
MOVE.B   (A0)+,D0       ; in D2 speichern.
BEQ      NOCHAR
```

```
CONV:

MOVE    D0,D1
SUBQ    #1,D1

CLP:

MOVE.B  (A0)+,D0
SUB.B   #'0',D0
BMI     NOVAL

CONV1:

CMP.B   #9,D0
BGT     NOVAL
MULU    #10,D2
ADD     D0,D2

NOVAL:

DBRA    D1,CLP

NOCHAR:

TST.L   D2
BNE     LL01
LEA     STRING2,A0

ERROR:

BSR     PRINT
BSR     WAIT
CLR     -(SP)
TRAP    #GEMDOS
****************************************


LL01:

MOVE    D2,D6
CMP     D7,D6           ; Genug Platz
                          vorhanden ?
BLE     OK              ; ja.
MOVE    D7,D6           ; sonst vorhandenen
                          Platz

OK:

MOVE    D6,D0
EXT.L   D0
LEA     ZIFF2,A0
BSR     WANDEL
PEA     INS01           ; Installation im
                          Super-Mode
MOVE    #SUPEXC,-(SP)
TRAP    #XBIOS
LEA     6(SP),SP
TST     D0              ; Fehler bei
                          Installation ?
BEQ     STAND           ; nein.
LEA     STRING4,A0
BRA     ERROR

STAND:

LEA     STRING3,A0
BSR     PRINT
BSR     WAIT
CLR     -(SP)           ; Keep Process
MULU    #1024,D6
MOVE.L  D6,LENGTH
ADD.L   #INSTALL-
        START+$120,D6
MOVE.L  D6,-(SP)
TRAP    #GEMDOS

****************************************

INS01:

MOVE.L  $B4,A0          ; Trapvector holen.
MOVE    TRAPNEU,D0      ; Spooler schon
                          installiert ?
CMP     (A0),D0
BEQ     INSRET          ; ja.
MOVE.L  A0,TRAPOLD      ; Alten Trapvector
                          #13 retten.
MOVE.L  #TRAPNEU,$B4    ; Neuen eintragen.
MOVE.L  VBLQUEUE,A0     ; VBL-Routine
                          eintragen.
MOVE.L  4(A0),VBLOLD
MOVE.L  #VBLROUT,4(A0)
CLR     D0

INSRET:

RTS

****************************************

WANDEL:

MOVE    #4-1,D5         ; 4 Ziffern in

INSLP:

DIVU    #10,D0          ; Wandeln in BCD.
SWAP    D0
ADD     #'0',D0         ; auf ASCII bringen.
MOVE.B  D0,0(A0,D5.W)   ; in den String
                          einfügen.
CLR     D0
SWAP    D0
DBF     D5,INSLP        ; 4 Ziffern.
RTS

****************************************

PRINT:

MOVE.L  A0,-(SP)
MOVE    #PRT,-(SP)
TRAP    #GEMDOS
LEA     6(SP),SP
RTS

****************************************
```

```
WAIT:

MOVE    #7,-(SP)
TRAP    #GEMDOS
LEA     2(SP),SP
RTS

****************************************

.DATA

STRING1:

DC.B    $1B,"E",$B,"p"
DC.B    $a,$d,"Jens Schuppener 1987"
DC.B    $1B,"q"
DC.B    $a,$a,$d,"Centronics Printer Spooler."
DC.B    $a,$d,"Ich habe noch"

ZIFF:

DC.B    "XXXX KByte Speicherplatz."
DC.B    $a,$d,"Wieviel KByte ?",0

STRING2:

DC.B    $A,$D,"Spooler nicht installiert.",0

STRING3:

DC.B    $A,$D,"Spooler mit"

ZIFF2:

DC.B    "XXXX KByte installiert !",0

STRING4:

DC.B    $1B,"E Spooler war bereits installiert !"
DC.     $A,$A,$D,"--- Taste drücken ---",0

INBUFF:

DC.B    4,0,"XXXX",0

.EVEN
```

Basiclisting Spooler in GFA-Basic.

```
' ------------ '
' Dateigenerator J.
' ------------ '
Open "O",#1,"\spool.tos"
Zeile=0
Sum=0
Do
Read A
Exit If A=-1
If A<10000
Sum=Sum+A
Out #1,A
Else
Flag=A-10000-Sum
Inc Zeile
Sum=0
Endif
Exit If Flag<>0
```

Apfel-männchen

SOFTWAREBESCHREIBUNG ZU 3D FRACTAL

Apfelmännchen, zauberhafte Bilder, mathematisch berechnet, kennt mittlerweile wohl jeder. Jede größere Computerzeitschrift berichtet ausführlich über diese abstrakten Gebielde und hat auch so manches Listing veröffentlicht.

Neu dürfte allerdings sein, daß man Apfelmännchen auch dreidimensional darstellen kann.

Das folgende Listing zeigt eine Version für den Atari ST mit Colormonitor und wurde in GFA-Basic erstellt.

Doch vorerst noch eine kurze Beschreibung zum Gebrauch dieses Programmes. Sie können es selbstverständlich nur laufen lassen, sofern Sie den GFA-Basic Interpreter besitzen.

Bevor Sie es starten, legen Sie bitte eine doppelseitig formatierte Diskette in Ihr Laufwerk ein.

3D FRACTAL generiert nun selbsttätig dreidimensionale fraktale Grafiken und speichert diese auf Diskette ab.

Da GFA-Basic pro Bild ca. 30 Minuten rechnet, ist es empfehlenswert, das Programm zu compilieren.

Mit „Bload ‚frac1.pic',1015808" laden Sie das erste Bild. Mittels einer kleinen Laderoutine können Sie die Bilder der Reihe nach anschauen.

```
Neu:
Setcolor 0,73
Cls
For T=1 To 10000
Next T
Xm=550
Ym=100
Xc=Rnd(0)
Yc=Rnd(0)
T=Int(Rnd(1)*30)
S=Int(Rnd(1)*100)
Mi=0
X1=-0.15
Xr=0.26
Yo=0.47
Yu=0.9
Goto Bild
A1:
Bsave "frac1.pic",1015808,32768
Goto Neu
A2:
Bsave "frac2.pic",1015808,32768
Goto Neu
A3:
Bsave "frac3.pic",1015808,32768
```

```
Goto Neu
A4:
Bsave "frac4.pic",1015808,32768
Goto Neu
A5:
Bsave "frac5.pic",1015808,32768
Goto Neu
A6:
Bsave "frac6.pic",1015808,32768
Goto Neu
A7:
Bsave "frac7.pic",1015808,32768
Goto Neu
A8:
Bsave "frac8.pic",1015808,32768
Goto Neu
A9:
Bsave "frac9.pic",1015808,32768
Goto Neu
A10:
Bsave "frac10.pic",1015808,32768
Goto Neu
A11:
Bsave "frac11.pic",1015808,32768
Goto Neu
A12:
Bsave "frac12.pic",1015808,32768
Goto Neu
A13:
Bsave "frac13.pic",1015808,32768
Goto Neu
A14:
Bsave "frac14.pic",1015808,32768
Goto Neu
A15:
Bsave "frac15.pic",1015808,32768
Goto Neu
A16:
Bsave "frac16.pic",1015808,32768
Goto Neu
A17:
Bsave "frac17.pic",1015808,32768
Goto Neu
A18:
Bsave "frac18.pic",1015808,32768
Goto Neu
A19:
Bsave "frac19.pic",1015808,32768
Goto Neu
A20:
Bsave "frac20.pic",1015808,32768
Goto Neu
Bild:
Dx=(Xr-Xl)/Xm
Dy=(Yu-Yo)/Ym
For P=0 To Ym
Y1=Yo+N*Dy
For I=0 To Xm
X=X1+M*Dx
Y=Y1
K=0
Label1:
X2=X^2
Y2=Y^2
Y=2*X*Y-Yc
X=X2-Y2-Xc
K=K+1
If (K<T) And (X2+Y2<S) Then
Goto Label1
Endif
U=M+53-N/2
U1=U+1
V=N+80
V1=V-3*(K-1)
Color 3
Line U+Mi,V,U+Mi,V1
Color 0
Line U1+Mi,V,U1+Mi,V1
Color 1
Line U+Mi,V1,U1+Mi,V1
Next I
Next P
Z=Z+1
If Z=1 Then
Goto A1
Endif
If Z=2 Then
Goto A2
Endif
If Z=3 Then
Goto A3
Endif
If Z=4 Then
Goto A4
Endif
If Z=5 Then
Goto A5
Endif
If Z=6 Then
Goto A6
Endif
If Z=7 Then
Goto A7
Endif
If Z=8 Then
Goto A8
Endif
If Z=9 Then
Goto A9
Endif
If Z=10 Then
Goto A10
Endif
If Z=11 Then
Goto A11
Endif
```

ATARI SPECIAL KOMMT JETZT DIREKT INS HAUS! SIEBENMAL IM JAHR FÜR 60 DM! SIE SPAREN ÜBER 20 PROZENT!

Finden Sie Ihre ATARI SPECIAL nicht immer am Kiosk? Vielleicht, weil schon ausverkauft? Möchten Sie ATARI SPECIAL schon vor der Kioskbelieferung in Händen haben? Dann gibt es jetzt eine Möglichkeit! Wir beliefern Sie im Abonnement mit sechs plus einer Ausgabe für ganze 60 DM (Inland) oder 80 DM (Ausland). Sie erhalten dann das jeweils druckfrische Heft, in der Regel sogar früher, als es am Kiosk hängt (so die Bundespost will). Sechsmal. Und außerdem gehört zum Abo noch unser jährlicher ATARI SPECIAL Sammelband im Wert von DM 14,80. Einzige Bedingung: Das Abo muß zum Zeitpunkt des Erscheinens dieses Bandes noch bestehen und bezahlt sein. Ist das ein Angebot?

WICHTIGE RECHTLICHE GARANTIE!

Sie können diesen Abo-Auftrag binnen einer Woche nach Zugang der Abo-Bestätigung durch den Verlag an Sie widerrufen. Es genügt die rechtzeitige Absendung. Bitte, bestätigen Sie durch Ihre zweite Unterschrift, daß Sie von diesem Widerspruchsrecht Kenntnis genommen haben.

ACHTUNG: WICHTIGER HINWEIS!

Sie können dieses Abonnement jeweils mit einer Frist von einem Monat zum Ende des Bezugszeitraumes (sechs Hefte) kündigen. Unterlassen Sie diese Kündigung, wird die Belieferung mit weiteren sechs Heften zu den gleichen Bedingungen fortgesetzt! Die Lieferung beginnt nach Eingang der Abo-Gebühr.

ABO-SERVICE-KARTE

Ich nehme zur Kenntnis daß die Belieferung erst beginnt, wenn die Abo-Gebühr dem Verlag zugegangen ist!

Ja, ich möchte von Ihrem Angebot Gebrauch machen.
Bitte senden Sie mir bis auf Widerruf ab sofort jeweils die nächsten sechs (+ 1) Ausgaben an untenstehende Anschrift. Wenn ich nicht vier Wochen vor Ablauf kündige, läuft diese Abmachung automatisch weiter.

Name _____
Vorname _____
Straße/Hausnr. _____
PLZ/Ort _____
Ich bezahle:
☐ per beiliegendem Verrechnungsscheck
☐ gegen Rechnung
☐ bargeldlos per Bankeinzug von meinem Konto
bei (Bank) und Ort _____
Kontonummer _____
Bankleitzahl _____
(steht auf jedem Kontoauszug)

Unterschrift _____
Von meinem Widerspruchsrecht habe ich Kenntnis genommen.

Unterschrift _____

ATARI SPECIAL II/88
ABO-SERVICE
POSTFACH 1161
D-8044
UNTERSCHLEISSHEIM

BIG-ASS
ATARI SOFTWARE SERVICE!

Hiermit bestelle ich in Kenntnis Ihrer Verkaufsbedingungen die Listings dieses Heftes auf

☐ Diskette (30 DM)
Name _____
Vorname _____
Straße/Hausnr. _____
PLZ/Ort _____
Ich bezahle:
☐ per beiliegendem Verrechnungsscheck / Bargeld
☐ bargeldlos per Bankeinzug von meinem Konto (nur möglich in der Bundesrepublik!)
bei (Bank) und Ort _____
Kontonummer _____
Bankleitzahl _____ (steht auf jedem Kontoauszug)
Verkaufsbedingungen: Lieferung nur gegen Vorkasse oder Bankabbuchung. Keine Nachnahme.
Umtausch bei Nichtfunktionieren.

Unterschrift _____
Bitte ausschneiden und einsenden an

ATARI LISTING EXTRA II/88
DISK-SERVICE
POSTFACH 1161
D-8044 UNTERSCHLEISSHEIM

ANZEIGENSERVICE

Die große Börse für jeden Zweck in ATARI-SPECIAL. Kostenlos für Privat-Inserenten. Spottbillig für gewerbliche Anbieter. Einfach Coupon ausschneiden, fotokopieren o.ä., ausfüllen und ab die Post -- Freimachen nicht vergessen! – Unsere Adresse steht auf dem Coupon, ebenso die Preise für gewerbliche Anbieter! Achtung! Wir weisen ausdrücklich darauf hin, daß wir offensichtlich gewerbliche Anzeigen nicht kostenlos veröffentlichen und uns jedweden Abdruck kostenloser Anzeigen vorbehalten müssen, insbesondere, wenn deren Inhalt gegen geltendes Recht verstößt. Private Chiffreanzeigen werden nicht aufgenommen. Für Privatanbieter: etwa bis zu acht Zeilen a 28 Anschläge. Für gewerbliche Anbieter: 5 DM p. Zeile mit 28 Anschläge.

ATARI SPECIAL
ANZEIGENABTLG
POSTFACH 1161
D-8044 UNTERSCHLEISSHEIM

Name _____
Vorname _____
Straße/Hausnr. _____
PLZ/Ort _____

DAS SONDERANGEBOT: PRIVATE KLEINANZEIGEN SIND KOSTENLOS

Das bietet Ihnen ab sofort ATARI SPECIAL: KLEINANZEIGEN SIND KOSTENLOS FÜR PRIVATANBIETER! Suchen Sie etwas, haben Sie etwas zu verkaufen, zu tauschen, wollen Sie einen Club gründen? Coupon ausfüllen, auf Postkarte kleben oder in Briefumschlag stecken und abschicken. So einfach geht das. Wollen Sie das Heft nicht zerschneiden, können Sie den Coupon auch fotokopieren. Oder einfach den Anzeigentext uns so schicken, auf Postkarte oder im Brief. Aber bitte mit Druckbuchstaben oder in Schreibmaschinenschrift!
Und: Einschließlich Ihrer Adresse und/oder Telefonnummer sollten acht Zeilen a 28 Anschläge nicht überschritten werden.
ACHTUNG: WICHTIGER HINWEIS!
Wir veröffentlichen nur Kleinanzeigen privater Inserenten kostenlos, gewerbliche Anzeigen kosten pro Zeile zu 28 Buchstaben DM 5,00 plus Mehrwertsteuer! Wir versenden für Privat-Inserenten keine Beleg-Exemplare!

DIE INSERTION IST NICHT VOM HEFTKAUF ABHÄNGIG! Chiffre-Anzeigen sind nicht gestattet! Wir behalten uns vor, Anzeigen, die gegen rechtliche, sittliche oder sonstige Gebote verstoßen, abzulehnen!
Anzeigenabdruck in der Reihenfolge ihres Eingangs, kein Rechtsanspruch auf den Abdruck in der nächsten Ausgabe!
Wir behalten uns vor, Anzeigen, die nicht zum Themenkreis des Heftes – Computer – gehören, nur insoweit zu berücksichtigen, wie es der Umfang des kostenlosen Anzeigenteils zuläßt.

VERDIENEN SIE GELD MIT IHREM ATARI ST

Haben Sie einen ATARI ST? Können Sie programmieren? In Basic oder Maschinensprache? Dann bietet ATARI SPECIAL Ihnen die Möglichkeit, mit diesem Hobby Geld zu verdienen.
Wie? Ganz einfach. Sie senden uns die Programme, die Sie für einen Abdruck als geeignet halten, zusammen mit einer Kurzbeschreibung, aus der auch die verwendete Hardware -- eventuelle Erweiterungen -- benutzte Peripherie – hervorgehen muß, ein.
Benötigt wird ein Datenträger mit dem Programm! Wenn die Redaktion sich überzeugt hat, daß dieses Programm läuft und sich zum Abdruck eignet, zahlen wir Ihnen pro abgedrucktem Programm, je nach Umfang, bis zu DM 500,–!

Sie erhalten Ihren Datenträger selbstverständlich zurück, wenn Sie einen ausreichend frankierten Rückumschlag mit Ihrer Adresse beifügen.

Bei der Einsendung müssen Sie mit Ihrer Unterschrift garantieren, daß Sie der alleinige Inhaber der Urheber-Rechte sind! Benutzen Sie bitte anhängendes Formular! (Wir weisen darauf hin, daß auch die Redaktion englische Fachzeitschriften liest und „umgestaltete" Programme ziemlich schnell erkennt).

Um Ihnen die Arbeit zu erleichtern, finden Sie hier ein Formular. Sie können es ausschneiden oder fotokopieren.

Name des Einsenders: _____
Straße/Hausnr./Tel.: _____
PLZ/Ort: _____

Hiermit biete ich Ihnen zum Abdruck folgende(s) Programm(e) an:

_____
_____
_____
_____

Benötigte Geräte: _____
_____

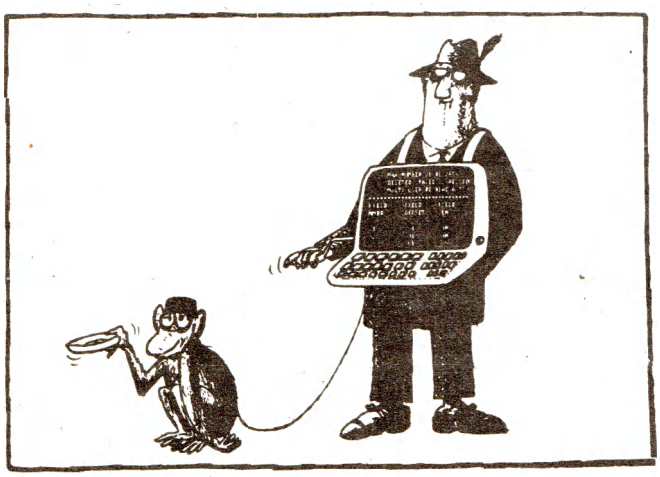
Beigefügt ☐ Listings ☐ Diskette

Ich versichere, der alleinige Urheber des Programms zu sein!
Hiermit ermächtige ich die Redaktion, dieses Programm abzudrucken und wirtschaftlich zu verwerten. Das Copyright geht an den Verlag über.

_____
Rechtsverbindliche Unterschrift

ATARI SPECIAL
Programm-Redaktion
Postfach 1161
D-8044 Unterschleißheim

```
If Z=12 Then
Goto A12
Endif
If Z=13 Then
Goto A13
Endif
If Z=14 Then
Goto A14
Endif
If Z=15 Then
Goto A15
Endif
If Z=16 Then
Goto A16
Endif
If Z=17 Then
Goto A17
Endif
If Z=18 Then
Goto A18
Endif
If Z=19 Then
Goto A19
Endif
If Z=20 Then
Goto A20
Endif
If Z=21 Then
Goto Fertig
Endif
Fertig:
Print "FERTIG"
End
```



```
DATA 0,0,1,36,6,134,0,0,2,84,47,6,63,60,10439
DATA 0,49,78,65,32,120,0,180,48,57,0,0,10629
DATA 0,4,176,80,103,0,0,38,35,200,0,0,1,10637
DATA 32,33,252,0,0,0,4,0,180,32,120,4,86,10743
DATA 35,232,0,4,0,0,1,28,33,124,0,0,0,148,10605
DATA 0,4,66,64,78,117,58,60,0,3,128,252,10830
DATA 0,10,72,64,6,64,0,48,17,128,80,0,66,10555
DATA 64,72,64,81,205,255,236,78,117,47,11219
DATA 8,63,60,0,9,78,65,79,239,0,6,78,117,10802
DATA 63,60,0,7,78,65,79,239,0,2,78,117,10788
DATA 27,69,27,112,10,13,74,101,110,115,10658
DATA 32,83,99,104,117,112,112,101,110,101,10971
DATA 114,32,49,57,56,55,27,113,10,10,13,10536
```

```
DATA 67,101,110,116,114,111,110,105,99,10933
DATA 115,32,80,114,105,110,116,101,114,10887
DATA 32,83,112,111,111,108,101,114,46,10,10828
DATA 13,73,99,104,32,104,97,98,101,32,110,10863
DATA 111,99,104,32,88,88,88,88,32,75,66,10871
DATA 121,116,101,32,83,112,101,105,99,104,10974
DATA 101,114,112,108,97,116,122,46,10,13,10839
DATA 87,105,101,118,105,101,108,32,75,66,10898
DATA 121,116,101,32,63,0,10,13,83,112,111,10762
DATA 111,108,101,114,32,110,105,99,104,10884
DATA 116,32,105,110,115,116,97,108,108,10907
DATA 105,101,114,116,46,0,10,13,83,112,10700
DATA 111,111,108,101,114,32,109,105,116,10907
DATA 32,88,88,88,88,32,75,66,121,116,101,10895
DATA 32,105,110,115,116,97,108,108,105,10896
DATA 101,114,116,32,33,0,27,69,32,83,112,10719
DATA 111,111,108,101,114,32,119,97,114,10907
DATA 32,98,101,114,101,105,116,115,32,105,10919
DATA 110,115,116,97,108,108,105,101,114,10974
DATA 116,32,33,10,10,13,45,45,45,45,45,10439
DATA 45,45,45,45,32,84,97,115,116,101,32,10757
DATA 100,114,129,99,107,101,110,32,45,45,10882
DATA 45,45,45,45,45,45,45,0,4,0,88,88,88,10583
DATA 88,0,42,0,0,0,42,26,6,14,6,10,8,12,10254
DATA 26,48,6,38,8,10,6,10,44,18,10,10,16,10250
DATA 52,32,10,22,10,20,24,12,6,14,6,10208,-1
Loop
Close #1
If A=-1
Print "Alle DataZeilen ok!
Else
Print "Datafehler in Data-Zeile No. ";Zeile
Endif
End
```

```
 DATA 96,26,0,0,2,134,0,0,1,10,0,0,0,0,0,10269
 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,96,0,1,50,10147
 DATA 65,239,0,6,48,23,8,0,0,13,102,0,0,10504
 DATA 4,78,104,48,40,0,0,12,64,0,3,102,0,10455
 DATA 0,10,48,40,0,2,103,0,0,10,32,121,0,10366
 DATA 0,1,32,78,208,34,120,4,162,48,31,51,10769
 DATA 0,35,31,72,225,31,31,33,201,4,162,10825
 DATA 32,57,0,0,1,40,176,185,0,0,1,36,110,10638
 DATA 0,255,242,18,40,0,5,67,249,0,0,1,52,10929
 DATA 36,57,0,0,1,44,19,129,40,0,82,185,10593
 DATA 0,0,1,40,82,130,180,185,0,0,1,36,111,10766
 DATA 0,0,4,66,130,35,194,0,0,1,44,112,255,10841
 DATA 34,120,4,162,76,217,248,248,47,25,11181
 DATA 63,25,33,201,4,162,78,115,74,185,0,10940
 DATA 0,1,40,103,0,0,108,8,56,0,0,250,1,10567
 DATA 102,0,0,98,64,198,0,124,7,0,67,248,10908
 DATA 136,0,18,188,0,7,16,17,0,0,0,128,19,10529
 DATA 64,0,2,18,188,0,15,34,57,0,0,1,48,10427
 DATA 65,249,0,0,1,52,16,48,24,0,19,64,0,10538
 DATA 2,18,188,0,14,16,17,0,0,0,32,19,64,10370
 DATA 0,2,2,0,0,223,19,64,0,2,70,198,83,10663
 DATA 185,0,0,1,40,82,129,178,185,0,0,1,10801
 DATA 36,111,2,66,129,35,193,0,0,1,48,74,10695
 DATA 185,0,0,1,28,103,0,0,10,32,121,0,0,10480
 DATA 1,28,78,208,78,117,0,0,0,0,0,0,0,0,10510
 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,42,10042
 DATA 79,42,109,0,4,46,45,0,4,4,135,0,0,10468
 DATA 1,52,142,252,4,0,4,71,0,128,48,7,72,10781
 DATA 192,65,249,0,0,2,208,97,0,0,248,65,11126
 DATA 249,0,0,2,134,97,0,1,10,72,121,0,0,10686
 DATA 3,136,63,60,0,10,78,65,79,239,0,6,10739
 DATA 65,249,0,0,3,137,66,130,16,24,103,10793
 DATA 0,0,34,50,0,83,65,16,24,4,0,0,48,107,10431
 DATA 0,0,16,12,0,0,9,110,0,0,8,196,252,10603
 DATA 0,10,212,64,81,201,255,230,74,130,11257
 DATA 102,0,0,20,65,249,0,0,2,251,97,0,0,10786
 DATA 188,97,0,0,198,66,103,78,65,60,2,188,11045
 DATA 71,111,0,0,4,60,7,48,6,72,192,65,249,10885
 DATA 0,0,3,38,97,0,0,128,72,121,0,0,2,26,10487
 DATA 63,60,0,38,78,78,79,239,0,6,74,64,10779
 DATA 103,0,0,12,65,249,0,0,3,63,96,0,255,10846
 DATA 190,65,249,0,0,3,24,97,0,0,114,97,10839
 DATA 0,0,124,66,103,204,252,4,0,35,198,10986
```

Maschinenroutine zur Eingabe von Datum und Uhrzeit beim Systemstart

Wie Sie sicherlich bereits wissen, besitzt der ATARI ST eine eingebaute Software-Uhr, die bei jedem Systemstart (nach dem Einschalten oder nach RESET) neu eingestellt werden muß, da sie nicht gepuffert ist.

Mit Hilfe des Kontrollfeldes kann der User sowohl das Datum wie auch die Uhrzeit ändern. Dazu ist es jedoch notwendig, daß sich das Accessory-File mit dem Kontrollfeld beim Booten des Systems auf der eingelegten Diskette befindet. Das hier abgedruckte Maschinenprogramm erlaubt die Eingabe von Datum und Uhrzeit. Es bietet gegenüber der Kontrollfeld-Alternative mehrere große Vorteile. Das Programmfile wird – in einem AUTO-Ordner abgelegt – beim Systemstart automatisch gestartet, so daß die Eingabe des aktuellen Datums und der momentanen Uhrzeit nicht vergessen werden kann. Weiterhin kann auf das Kontrollfeld-Accessory verzichtet werden, das recht viel Speicherplatz in Anspruch nimmt und trotzdem jederzeit die aktuellen Daten eingegeben werden.

Das Maschinenprogramm hat eine Länge von 461 Bytes und wurde mit Hilfe eines DATA–Generators in den DATA-Zeilen eines BASIC-Programms abgelegt. Durch dieses Verfahren ist gewährleistet, daß jedermann das Maschinenprogramm eingeben kann, selbst wenn er keinen Assembler besitzt. Außerdem nimmt ein BASIC-Listing weitaus weniger Platz in Anspruch als der Quelltext, so daß man auch Zeit bei der Eingabe spart. Interessierte Leser erhalten auf Wunsch ein Listing des Quelltextes gegen einen ausreichend frankierten Rückumschlag beim Programmautor. Die Adresse finden Sie am Ende des Artikels.

Zur Bedienung des BASIC-Programms: Geben Sie den Programmtext zunächst im ST-BASIC-Editor ein und speichern Sie das Programm sicherheitshalber ab. Anschließend können Sie es mit RUN starten. Der Rechner fragt nach dem Namen der Datei, in der das Maschinenprogramm abgelegt werden soll. Geben Sie einen Namen mit geeigneter Extension (PRG oder TOS) ein. Es handelt sich um ein TOS-Programm. Wenn es in einem AUTO-Ordner abgelegt werden soll, so muß die Endung jedoch „PRG“ lauten, damit es automatisch gestartet wird.

Der Computer wandelt die DATA-Zeilen dann automatisch in ein Maschinenprogramm um und speichert dieses auf der Diskette ab. Fehlerhafte DATA-Werte, die anhand einer Checksumme am Ende jeder Zeile erkannt werden, gibt der Computer auf dem Bildschirm aus, so daß eine korrekte Eingabe gewährleistet ist.

Wenn das Maschinenprogramm später gestartet wird (durch Anklicken oder automatisch beim Systemstart), so muß der User zunächst das aktuelle Datum eingeben. Der Rechner zeigt die gewünschte Form an (TT-MM-JJJJ). Ein gültiges Datum wäre beispielsweise „10-01-1987“. Das Programm überprüft automatisch, ob die eingegebenen Werte im richtigen Bereich liegen und verlangt bei Fehlern eine neue Eingabe.

Nach der Eingabe des Datums folgt die Eingabe der Uhrzeit, wobei man sich an die Form „ss:mm:ss“ halten sollte. Eine korrekte Angabe ist also zum Beispiel „21:33:14“. Die Sekundenangabe sollte ein gerader Wert sein, da der ATARI ST nur alle zwei Sekunden die Uhr per Interrupt weiterstellt und deshalb nur gerade Sekunden verwendet. Auch bei der Zeitangabe werden die Werte auf den richtigen Bereich geprüft.

Es gibt übrigens die Möglichkeit, auf die Eingabe der Daten zu verzichten, indem man bei der entsprechenden Abfrage nur die RETURN-Taste betätigt. Die intern gespeicherten Werte bleiben dann unverändert erhalten.

Die Eingaben selbst können mit Hilfe der Backspace-Taste korrigiert werden, sofern man noch nicht RETURN gedrückt hat.

Wenn beide Eingaben korrekt vorgenommen wurden, kehrt das Programm automatisch zum Desktop zurück.

Bitte beachten Sie, daß das Programm nur dann in einem AUTO-Ordner eingesetzt werden sollte, wenn sich das Kontrollfeld-Accessory nicht auf dieser Diskette befindet, da dieses die Uhrzeit nach der Eingabe mit Hilfe des Maschinenprogramms erneut verändert.

Die Adresse, bei der Sie den Quelltext erhalten können, lautet: Oliver Steinmeier, Mauerstraße 1, 4934 Horn-Bad Meinberg 1.

Oliver Steinmeier

```
100 REM ****************************************
110 REM *                                      *
120 REM *        Datums- und Zeiteingabe       *
130 REM *           für den ATARI ST           *
140 REM *             BASIC-LOADER             *
150 REM *                                      *
160 REM *                                      *
170 REM *                                      *
180 REM ****************************************
190 REM
200 FULLW 2 : CLEARW 2
210 GOTOXY 15,2
220 PRINT "Datumseingabe"
230 GOTOXY 2,5
240 INPUT "Filename für die Programmdatei : ",
250 GOTOXY 2,7
```

```
260 INPUT "Diskette einlegen und RETURN drücken!",IP$
270 GOTOXY 15,10 : PRINT "Bitte warten"
280 REM
290 REM ------ DATAs einlesen -------
300 REM
310 ZEILE = 600 : REM 600 = erste DATA-Zeile
320 READ LAENGE
330 DIM F%(LAENGE/2+1)
340 START = VARPTR(F%(0))
350 FOR I=0 TO LAENGE/2 STEP 7
360 SUMME%= 0 : REM Checksumme löschen
370 ZEILE = ZEILE + 10
380 FOR J=I TO I+6
390 READ WERT%
400 SUMME%= SUMME%+ WERT%
410 F%(J)= WERT%
420 NEXT J
430 READ CHECK%
440 IF CHECK%<> SUMME%THEN 500
450 NEXT I
460 GOTOXY 12,15
470 PRINT "Keine Fehler in den DATAs"
480 BSAVE F$,START,LAENGE
490 END
500 REM ----- Fehler ------
510 GOTOXY 12,15
520 PRINT "Fehler in Zeile";ZEILE;"!!!"
530 GOTOXY 12,17
540 INPUT "Bitte RETURN drücken !!!",IP$
550 END
560 REM
570 REM ------- DATA-Zeilen -------
580 REM
600 DATA 461
610 DATA 24602,0,324,0,92,0,12,25030
620 DATA 0,0,0,0,0,0,0,0
630 DATA 16188,1,16188,21,20046,22671,11388,20967
640 DATA 0,324,24832,240,11388,0,362,-28390
650 DATA 24832,230,24832,272,5116,11,0,-10243
660 DATA 416,18553,0,416,16188,10,20033,-9920
670 DATA 23695,19001,0,417,26368,66,11388,15399
680 DATA 0,418,24832,196,3141,31,28344,-8574
690 DATA 10245,24832,184,3141,12,28332,-5307,-4097
700 DATA -10171,24832,170,3141,1980,28062,1093,-16429
710 DATA 1980,5692,9,-6299,-10171,16132,16188,23531
720 DATA 43,20033,22671,11388,0,386,24832,13817
730 DATA 120,24832,162,5116,9,0,416,30655
740 DATA 18553,0,416,16188,10,20033,23695,13359
750 DATA 19001,0,417,26368,66,11388,0,-8296
760 DATA 418,24832,86,3205,0,23,28352,-8620
770 DATA 10245,-4796,24832,70,3205,0,59,-31921
780 DATA 28336,-10171,-5308,24832,54,3205,0,-24588
790 DATA 59,28320,-7611,-10171,16132,16188,45,-22574
800 DATA 20033,22671,16188,0,16188,21,20046, 29611
810 DATA 22671,16188,0,20033,12046,16188,9,21599
820 DATA 20033,23695,20085,17029,7198,3078,48,25630
830 DATA 27904,22,3078,57,28160,14,1094,-5207
840 DATA 48,-13572,10,-9658,24802,20085,11388,-32433
```

```
850 DATA 0,416,16926,-16900,0,429,28150,29021
860 DATA 20085,6981,7001,8506,17440,24864,29728,-16467
870 DATA 29984,27936,29472,25888,26912,28192,26400,-1824
880 DATA 24864,25120,25888,14858,3338,3328,8224,-25452
890 DATA 17505,29813,27936,10324,21549,19789,11594,7438
900 DATA 19018,18985,8250,8192,2573,2573,8224,2279
910 DATA 23141,26996,8232,26728,14957,27962,29555,26499
920 DATA 10528,8224,8250,8192,0,0,14,-30328
930 DATA 2576,1550,2624,4102,3594,-32760,0,-18314
```

Senso

Dieses beliebte Spiel soll Ihnen helfen, Ihr Gedächtnis zu trainieren. Können Sie den Computer schlagen? Töne, die der Rechner spielt, müssen per Anwahl durch einen Mausklick in der richtigen Reihenfolge wiederholt werden. Die Melodien werden immer komplexer. Dadurch verliert das Spiel auch nach längerer Spielzeit nichts von seinem Reiz. Der User muß schon sehr auf Draht sein, um seinen Computer zu schlagen. Dieses Programm ist in GFA-Basic geschrieben.

```
' BENUTZTE VARIABLEN
'
' T,TT,TC SCHLEIFEN
' ZA       ZÄHLT ANZAHL DER DURCHGÄNGE BEI ABFRAGE
' ZAEHLER ZÄHLT    ''     ''      ''       '' ZEIGEN
' X,Y,K   MOUSE STATUS
' ZUFALL  ENTHÄLT AUSWAHL DES NÄCHSTEN FELDES
' A       RETTET T
' R       FARBREGISTER BEI PROCEDURE WAIT(REGISTER)
' SCORE   PUNKTE ANZAHL
'
' BENUTZTE FELDER
'
' MERKER()   ENTHÄLT REIHENFOLGE DER FELDER
'
'   S E N S O
On Break Gosub Reset
Dim Merker(100)
'
' Farbwerte bestimmen
'
Setcolor 1,7,7,5      ! *
Setcolor 4,0,0,0      ! Schatten
Setcolor 5,7,0,0      ! feld 3
Setcolor 6,0,7,0      ! feld 1
Setcolor 7,0,0,7      ! feld 4
Setcolor 8,7,1,7      ! feld 5
Setcolor 3,7,7,4      ! feld 2
Setcolor 10,5,0,1
Setcolor 0,55         ! Hintergrund
'
' Bildschirm aufbauen
'
Deffill 4
```

```
Pbox 5,5,315,195
Deffill 11
Pbox 0,0,310,190
Color 4
Deftext 4,0,0,13
Graphmode 2
Text 130,20," SENSO "
Text 220,80," DURCHGÄNGE"
Text 20,80," SCORE"
Graphmode 1
For T=1 To 5
  Color 4
  Deffill 4,1,1
  Pbox 100+5,T*30+5,220+5,T*30+20+5
  Color 4+T
  Deffill 4+T,1,1
  Pbox 100,T*30,220,T*30+20
Next T
@Wait(0)
'
' Main Job (Hauptprogramm)
'
Do
  Zufall=Random(4)+1
  Inc Zaehler
  Merker(Zaehler)=Zufall
  Deftext 4,0,0,13
  Graphmode 1
  Text 270,100,Zaehler
  Text 40,100,Score
  Defmouse 2
  Pause 20
  @Show
  Defmouse 0
  @Ask
  If Zufall=Merker(Zaehler-1)
                      ! WENN FELD VORHIN (-1)
    Add Score,5       ! DANN 5
  Else
    Add Score,10      ! ANDENFALL 10 PUNKTE MEHR
  Endif
Loop
'
' Hauptprogramm ENDE
'
' Proceduren
'
Procedure Show
  For Tc=1 To Zaehler
    A=Merker(Tc)
    Restore
    For T=1 To A
      Read Y
    Next T
    Colo=Xbios(7,Y,-1) ! xbios(7,farbregister,-1)
    Setcolor Y,5,0,1   ! der farbwert des reg.
    Sound 1,15,A,3,2
    Vsync
```

SENSO

```
      Pause 20
      Sound 1,0,0,0
      Setcolor Y,Colo
      Pause 10
    Next Tc
  Return
  '
  Procedure Ask
    Za=0
    Do
     Mouse X,Y,K
      For T=1 To 5
       If X>100 And X<220 And Y>T*30 And Y<T*30+20 And K=1
          Inc Za
          @Klick
          @Uberwachung
       Endif
      Next T
      Exit If Za=Zaehler
    Loop
  Return
  '
  Procedure Klick
    A=T
    Restore
    For T=1 To A
      Read Y
    Next T
    Colo=Xbios(7,Y,-1)
    Setcolor Y,5,0,1
    Sound 1,15,A,3,2
    Vsync
    Pause 20
    Sound 1,0,0,0
    Setcolor Y,Colo
    Pause 10
  Return
  '
  Procedure Uberwachung
    If A<>Merker(Za)
      @Fehler
    Endif
  Return
  '
  Procedure Fehler
    For Tt=0 To 12
      For T=0 To 15
        Sound 1,15,Tt,T,1
      Next T
      Print At(40,40)
      Print
    Next Tt
    Deffill 4
    Pbox 105,55,225,155
    Deffill 9
    Pbox 100,50,220,150
    '  Deffill 0
    '  Pbox 120,70,200,130
```

```
   Graphmode 3
   Deftext 1Ø,Ø,Ø,13
   Text 115,1ØØ,"  G A M E  "
   Text 115,12Ø,"  O V E R  "
   Graphmode 1
   @Wait(8)
   Repeat
   Until Mousek=Ø
   Alert 1," ¦ ¦ NOCH 'N  GAME ??",1," JA ¦ NEIN ",A1
   If A1=1
     Run
   Else
     End
   Endif
Return
'

Procedure Reset
   Sound 1,Ø,Ø,Ø
   Setcolor Ø,55
   Setcolor 1,77
   Edit
Return
'

Procedure Wait(R)
   Hidem
   Mouse X,Y,K
   Do
     '
     '
     Setcolor R,1
     '
     '
     Setcolor R,2
     '
     Setcolor R,3
     '
     '
     Setcolor R,4
     '
     Setcolor R,5
     '
     '
     Setcolor R,6
     '
     Setcolor R,7
     '
     Setcolor R,8
     '
     Exit If X<>Mousex Or Y<>Mousey
   Loop
   Showm
   Setcolor Ø,5
Return
'

Data 6,3,5,7,8
```


SENSO

Dir-Lister

Das folgende Problem kennt wohl jeder ST-User; Man sucht ein bestimmtes Programm, weiß jedoch nicht, auf welcher Diskette es in welchem Subdirectory abgespeichert ist. Normalerweise muß nun jeder Ordner der Disketten einzeln nach der Datei durchsucht werden. Mit Hilfe des hier abgedruckten Assemblerprogramms DIR LISTER kann der USER sich alle auf einer Diskette gespeicherten Programme in einem Arbeitsgang anzeigen lassen. Die Ausgabe kann auf Wunsch auch auf einem Drucker erfolgen.

Die Bedienung des Programms ist sehr einfach. Nach dem Starten durch doppeltes Anklicken muß zunächst ein Suchfilename eingegeben werden. Bei der Ausgabe der Dateien auf der Diskette werden nun die entsprechenden Files berücksichtigt. Natürlich kann der Benutzer hier auch das „*" als Joker-Zeichen verwenden. Folgende Eingaben sind also zulässig:

*.BAS	zeigt alle Dateien mit der Extension „BAS" an
GRAFIK.*	zeigt alle Dateien mit dem Namen „GRAFIK" an
.	zeigt alle Dateien mit beliebigem Namen und beliebiger Extension an.

Drückt der User einfach nur die RETURN-Taste, so wird „*.*" als Default-Wert eingesetzt.

Es ist also möglich, sämtliche oder nur bestimmte Dateien einer Diskette ausgeben zu lassen. Nach dieser Eingabe fragt das Programm nach der Ausgabeeinheit. Sollen die Dateien auf dem Bildschirm angezeigt werden, so muß „0" eingegeben werden. Für die Ausgabe auf dem Drucker lautet die korrekte Antwort „1". Um das Programm zu beenden, muß hier lediglich die ESC-Taste gedrückt werden.

Schließlich verlangt das Programm noch die Eingabe des Laufwerks. Je nach Eingabewert (z.B. „A", „B", „C") wird das Directory der eingelegten Diskette, der RAM-Floppy oder der Festplatte angezeigt.

Das Programm zeigt – wenn vorhanden – zunächst den Namen der Diskette an. Anschließend werden sämtliche Subdirectories der Reihe nach bearbeitet. Alle enthaltenen Files werden, sofern es der eingegebene Suchfilename zuläßt, angezeigt oder ausgedruckt. Ein „*" vor dem Namen signalisiert, daß die Datei schreibgeschützt ist. Nach dem jeweiligen Dateinamen werden die Länge, das Datum der letzten Veränderung und die zugehörige Uhrzeit ausgegeben. Die abgedruckte Bildschirmhardcopy verdeutlicht sämtliche Ein- und Ausgaben des Programms.

Das Programm wird nach jedem Durchlauf neu gestartet. Ein Abbruch ist – wie bereits erwähnt – durch Drücken der ESC-Taste bei der zweiten Eingabe möglich.

Wie Sie dieser Beschreibung des Programms entnehmen konnten, gibt es zwei Anwendungsgebiete für den DIR-LISTER. Er kann bei der Suche nach einer bestimmten Datei behilflich sein und außerdem zur Ausgabe eines Verzeichnisses sämtlicher Dateien einer Diskette dienen. Somit kann der Benutzer sich sehr leicht eine Übersicht aller vorhandenen Dateien erstellen. Für Festplatten-Besitzer gilt natürlich entsprechendes. Das Programm wurde mit Hilfe des MCC-Assemblers erstellt und läuft unter TOS.

```
*****************************************
*                                       *
*              DIR-LISTER               *
*                 für                   *
*              ATARI ST                 *
*                                       *
*                                       *
*                                       *
*                                       *
*****************************************

          TEXT

GEMDOS    equ     $01   GEMDOS-Trap
PRTLINE   equ     $09   Textausgabe
PRTOUT    equ     $05   Textausgabe auf Drucker
READLINE  equ     $0A   Texteingabe
SETDTA    equ     $1A   DTABUFFER setzen
TERM      equ     $00   Programmende
CONIN     equ     $01   Eingabe eines Zeichens
CONOUT    equ     $02   Ausgabe eines Zeichens
SETDRV    equ     $0E   Festlegung des Laufwerks
SFIRST    equ     $4E   Filesuch-Funktion
SNEXT     equ     $4F   Suche weiteres File .
CHDIR     equ     $3B   CHDIR-Funktion
SETBLOCK  equ     $4A   Speicherblock reservieren
ESC       equ     $1B   ESC-Taste

     movea.l 4(sp),a5          Basepage-Adresse in a5
     lea     STACK+1000,sp     neue Stackadresse setzen
     movea.l 12(a5),a6         Textsegmentlänge +
     adda.l  20(a5),a6         Datasegmentlänge +
     adda.l  28(a5),a6         BSS-Segmentlänge +
     adda.l  #$100,a6          Basepagelänge auf
     move.l  a6,-(sp)          den Stack legen
     move.l  a5,-(sp)          Basepage-Adresse,
     move    #0,-(sp)          Dummywort und Funktions-
     move    #SETBLOCK,-(sp)   nummer SETBLOCK auf Stack
     trap    #GEMDOS           legen und GEMDOS aufrufen
     add.l   #12,sp            Stack aufräumen
     tst     d0                Fehler aufgetreten?
     bne     ENDE              ja -> ENDE
     pea     DTABUFF           Bufferadresse auf
     move    #SETDTA,-(sp)     den Stapel legen und
     trap    #GEMDOS           an GEMDOS übergeben.
     addq.l  #6,sp             Stack aufräumen

BEGINN
     lea     CURSON,a6         Cursor ein
     bsr     MONITOR
     lea     TITLE,a6          Kopfzeile ausgeben
     bsr     MONITOR
     lea     SUCHNAME,a6       Frage nach Filename auf
     bsr     MONITOR           dem Monitor ausgeben
     move.b  #12,FILEN1
     pea     FILEN1            Adresse des Eingabebuffers u.
     move    #READLINE,-(sp)   READLINE-Funktion auf Stack
     trap    #GEMDOS           GEMDOS aufrufen
```

```
      addq.l   #6,sp              Stack aufräumen
      tst.b    dØ                 Zeichen eingegeben ?
      bne.s    SET_FILE           ja -> Buffer setzen
      move.l   #$2A2E2AØØ,FILEN   *.* als Suchname setzen
      bra.s    GET_AUSG           Ausgabegerät auswählen

SET_FILE
      lea      FILEN+2,a6
      adda.w   dØ,a6   Länge der Eingabe addieren
      clr.b    (a6)    Endzeichen setzen

GET_AUSG
     lea       DRUCKER,a6        Frage nach dem
     bsr       MONITOR           Monitor ausgeben
     move      #CONIN,-(sp)
     trap      #GEMDOS
     addq.l    #2,sp
     cmpi.b    #ESC,dØ           ESC gedrückt ?
     beq       ENDE              ja -> ENDE
     subi.b    #48,dØ            ASCII von 'Ø' abziehen
     move.b    dØ,d5             in d5 abspeichern
     lea       PROMT,a6          Eingabeaufforderung
     bsr       MONITOR
     move      #CONIN,-(sp)      Eingabe eines Zeichens
     trap      #GEMDOS
     addq.l    #2,sp             Stack aufräumen
     andi.b    #7,dØ             Laufwerksnummer berechnen
     subq.b    #1,dØ
     move      dØ,-(sp)          Drive-Nr. auf Stack
     move      #SETDRV,-(sp)     GEMDOS-Funktion SETDRV
     trap      #GEMDOS           aufrufen
     addq.l    #4,sp             Stack aufräumen

     move.l    #$5C5CØØØØ,BUFFER   Hauptdirectory
     bsr       CHNGEDIR          aktivieren
     move      #8,-(sp)          Attribut 8: Diskettenname
     move.l    #FILE,-(sp)       Name: *.*
     move      #SFIRST,-(sp)     GEMDOS-Funktion SFIRST
     trap      #GEMDOS           aufrufen und
     addq.l    #8,sp             Stack aufräumen
     tst.b     dØ                Fehler aufgetreten?
     bne       NOVOLUME          ja, kein Diskname
     lea       DISKNAME,a6       "Diskname" ausgeben
     bsr       AUSGABE
     clr.b     DTABUFF+44        Endmarkierung setzen
     lea       DTABUFF+3Ø,a6     Namen der Diskette
     bsr       AUSGABE           ausgeben
NOVOLUME
     bsr       CRLF              anschließend CR & LF
     clr.l     d1                EBENE (d1) initialisieren
     lea       BUFFER+1,a1       Adr. d. PATH-Buffers in a1
     clr.b     d2                Zahl der Subdirs in d2
START
      bsr.s    FILEAUSG
START1
      bsr      GETDIR             Neuen Subdir-Namen holen
      tst.b    dØ                 Überprüfen, ob Subdir
      bne.s    NODIR              kein Subdir -> zurück
```

```
        move.l  a1,-(sp)        alte Endadresse auf Stack
        move.b  d3,-(sp)        neue Subdir-Zahl auf Stack
        addq.b  #1,d1           EBENE erhöhen
        bsr     SETPATH         neuen Pfad setzen
        move.b  #2,d2
        bra.s   START           neue Namen ausgeben
NODIR
        tst.b   d1              EBENE = Ø
        beq.s   ENDE1           Hauptdirectory erreicht
        move.b  (sp)+,d2        alte Subdir-Zahl holen
        movea.l (sp)+,a1        alte Endadresse holen
        clr.b   (a1)            Null ans Stringende
        bsr     CHNGEDIR        Ins vorige Dir zurück
        subq.b  #1,d1           EBENE verringern
        bra.s   START1          und neues Subdir suchen
ENDE1
        lea     TASTE,a6        Ausgabe der Meldung
        bsr     MONITOR         "Taste drücken"
        move    #CONIN,-(sp)    Funktion CONIN aufrufen
        trap    #GEMDOS         Tastendruck erwarten
        addq.l  #2,sp           Stack aufräumen
        bra     BEGINN          Neustart
ENDE
        lea     CURSOFF,a6      Cursor ausschalten
        bsr     MONITOR
        move    #TERM,-(sp)     Programm beenden
        trap    #GEMDOS
FILEAUSG
        bsr     CRLF
        lea     BUFFER,a6       Adresse des PATH-Buffers
        bsr     AUSGABE         PATH ausgeben
        bsr     CRLF
        movea.l #FILEN,a6       Adr. des Filenamens in a6
        bsr     SRCHFRST        Suche ersten Filenamen
        tst.b   dØ              prüfen, ob File vorhanden
        bne.s   ZURUECK         kein File -> Subdir.
        cmp.b   #$1Ø,DTABUFF+21 Überprüfung, ob Subdir.
        beq.s   NXTFILE         nächstes File suchen
        bsr     FILEOUT         File vorhanden -> ausgeben
NXTFILE
        bsr     SRCHNXT         Suche nächstes File
        tst.b   dØ              prüfen, ob File vorhanden
        bne     ZURUECK         kein File -> Subdir.
        cmp.b   #$1Ø,DTABUFF+21 Überprüfung, ob Subdir.
        beq.s   NXTFILE         nächstes File suchen
        bsr     FILEOUT         File vorhanden -> ausgeben
        bra.s   NXTFILE         nächstes File suchen
ZURUECK
        rts                     zurück zum Hauptprogramm

GETDIR
        clr.l   d3              Zähler löschen
        movea.l #FILE,a6        Adresse von *.* in a6
        bsr     SRCHFRST        Suche erstes Subdir.
        tst.b   dØ              Überprüfung, ob vorhanden
        bne.s   ZURUECK         kein File/Subdir.
        cmp.b   #$1Ø,DTABUFF+21 Überprüfung, ob Subdir
        bne.s   NXTDIR1         weitersuchen
```

```
      addq.b   #1,d3              Zähler erhöhen
NXTDIR
      cmp.b    d2,d3              Subdir erreicht ?
      bgt.s    ZURUECK            ja, zurück
NXTDIR1
      bsr      SRCHNXT            Suche nächstes Subdir.
      tst.b    dØ                 Überprüfung, ob vorhanden
      bne.s    ZURUECK            kein File/Subdir -> zurück
      cmp.b    #$1Ø,DTABUFF+21    Überprüfung, ob Subdir
      bne.s    NXTDIR             Weitersuchen
      addq.b   #1,d3              Zähler erhöhen
      bra.s    NXTDIR             Überprüfen

SETPATH
      lea      DTABUFF+3Ø,a2      Anfang des Namens
MOVELP
      move.b   (a2)+,(a1)+        Übertragung des
      tst.b    (a2)               Subdirectory-Namens
      bne      MOVELP             in den Pfad-Buffer
      move.b   #'\',(a1)+
      clr.b    (a1)
      bsr      CHNGEDIR           Neues Subdir. aktivieren
      rts

SRCHFRST
      move     #$1Ø,-(sp)         Attribut: Subdirectory
      move.l   a6,-(sp)           Filename auf Stack
      move     #SFIRST,-(sp)      GEMDOS-Funktion SFIRST
      trap     #GEMDOS            aufrufen und
      addq.l   #8,sp              Stack aufräumen
      rts

SRCHNXT
      move     #SNEXT,-(sp)       GEMDOS-Funktion SNEXT
      trap     #GEMDOS            aufrufen und
      addq.l   #2,sp              Stack aufräumen
      rts

FILEOUT
      cmp.b    #1,DTABUFF+21      File schreibgeschützt?
      beq.s    READONLY           ja -> Ausgabe
      move     #3,d6              drei Leerzeichen
      bsr      SPACELP            ausgeben
      bra.s    FILEOUT1           Filename ausgeben
READONLY
      lea      ROTEXT,a6          READ-ONLY-String
      bsr      AUSGABE            ausgeben.

FILEOUT1
      clr.b    DTABUFF+44         Endmarkierung
      lea      DTABUFF+3Ø,a6
      bsr      AUSGABE
      move     #2Ø,d6             Zahl der Zeichen bis Länge
      sub      d7,d6              Länge des Filenamens subtr.
      move     d6,-(sp)           und auf den Stack legen
      move.l   DTABUFF+26,dØ
      bsr      HEXDEZ             Umrechnung ins Dez.-System
      move     (sp)+,d6           Spacelänge vom Stack holen
```

```
        sub.l   a3,d6           Länge der Zahl subtrahieren
        bsr     SPACELP         Leerzeichen ausgeben
        move.l  a2,a6           Anfangsadresse der Zahl an
        bsr     AUSGABE         a6 übergeben, dann Wert
        move    #4,d6           4 Leerzeichen
        bsr     SPACELP         ausgeben
        move.b  DTABUFF+25,dØ   Datum laden und Bit
        andi.l  #$1F,dØ         Ø bis 4 isolieren
        bsr.s   TAGMONAT        Tag ausgeben
        move    DTABUFF+24,dØ   Datum laden und Bit
        andi.l  #$Ø1EØ,dØ       5 bis 8 isolieren
        asr     #5,dØ           Monatsbits verschieben und
        bsr.s   TAGMONAT        ausgeben
        move    DTABUFF+24,dØ   Datum laden und Bit
        andi.l  #$FEØØ,dØ       9 bis 15 isolieren
        asr     #8,dØ           Monatsbits verschieben
        asr     #1,dØ
        addi    #198Ø,dØ        Basisjahr addieren
        bsr     HEXDEZ          ins Dezimalsystem wandeln
        movea.l a2,a6           Anfangsadresse übergeben
        bsr     AUSGABE         und Zahl ausgeben
        move    #4,d6           4 Leerzeichen
        bsr     SPACELP         ausgeben
        move.b  DTABUFF+22,dØ   Uhrzeit laden und
        andi.l  #$F8,dØ         Bit 11 bis 15 isolieren
        asr     #3,dØ           Stundenbits verschieben
        bsr.s   STDMIN          Stundenzahl ausgeben
        move    DTABUFF+22,dØ   Uhrzeit laden und
        andi.l  #$Ø7EØ,dØ       Bit 5 bis 1Ø isolieren
        asr     #5,dØ           Minutenbits verschieben
        bsr.s   STDMIN          Minutenzahl ausgeben
        move.b  DTABUFF+23,dØ   Uhrzeit laden und
        andi.l  #$1F,dØ         Bit Ø bis 4 isolieren
        asl.b   #1,dØ           Sekundenzahl verdoppeln
        bsr.s   HEXDEZ          und in Dez. umwandeln
        cmpa    #1,a3           Länge = 1 Zeichen ?
        bne.s   SEKAUSG         nein -> ausgeben
        subq.l  #1,a2           2 Zeichen ausgeben
SEKAUSG
        movea.l a2,a6           Adresse übergeben und
        bsr     AUSGABE         Sekundenzahl ausgeben
        bsr     CRLF            und CR/LF ausgeben.
        rts

TAGMONAT
        bsr.s   HEXDEZ          Umrechnung ins Dez-System
        cmpa    #1,a3           Länge = 1 Zeichen ?
        bne.s   TGMONAT1        nein -> ausgeben
        subq.l  #1,a2           2 Zeichen ausgeben
TGMONAT1
        movea.l a2,a6
        move    #$2EØØ,DEZZAHL+1Ø  '.' und $ØØ ans Ende
        bsr     AUSGABE         und Wert ausgeben
        rts

STDMIN
        bsr.s   HEXDEZ          Umrechnung ins Dez-System
        cmpa    #1,a3           Länge = 1 Zeichen ?
```

```
      bne.s   STDMIN1           nein -> ausgeben
      subq.l  #1,a2             2 Zeichen ausgeben
STDMIN1
      movea.l a2,a6
      move    #$3A00,DEZZAHL+10  ':' und $00 ans Ende
      bsr.s   AUSGABE           und Wert ausgeben
      rts

CHNGEDIR
      pea     BUFFER            Pfadadresse auf Stack
      move    #CHDIR,-(sp)      Aufruf der GEMDOS-
      trap    #GEMDOS           Funktion CHDIR
      addq.l  #6,sp             Stack aufräumen
      rts

HEXDEZ
      lea     DEZZAHL,a6        Bufferadresse laden
      moveq   #9,d6             Stellenzähler laden
      lea     TABELLE,a0        Tabellenadresse nach a7
HEXDEZLP
      move.l  (a0)+,d4          Stellenwert laden
      moveq   #0,d7             Zähler auf 0 setzen
HXDEZLP1
      addq.b  #1,d7
      sub.l   d4,d0             Tabellenwert subtrahieren
      bge.s   HXDEZLP1
      subq.b  #1,d7             Zähler wurde zu oft erhöht
      add.l   d4,d0             Wert wieder erhöhen
      add.b   #$30,d7           Zähler auf ASCII-Format
      move.b  d7,(a6)+
      dbmi    d6,HEXDEZLP
      clr.b   (a6)              Endzeichen setzen
      lea     DEZZAHL,a2        Bufferadresse laden
      movea.l a6,a3           Ende des Zahlenstrings laden
ZEROLP
      cmp.b   #'0',(a2)+        Nullen am Anfang der Zahl
      beq.s   ZEROLP            löschen
      subq.l  #1,a2           a6 auf Anfang der Zahl setzen
      tst.b   (a2)            zeigt a6 auf $00 ?
      bne.s   NOZERO            nein -> Länge bestimmen
      subq.l  #1,a2
NOZERO
      suba.l  a2,a3             Länge bestimmen
      rts

SPACELP
      subq    #1,d6             Zahl vermindern
SPACELP1
      lea     SPACE,a6          Adresse von Space in a6
      bsr.s   AUSGABE           Ausgabe des Leerzeichens
      dbra    d6,SPACELP1       Wiederholung bis < 0
      rts

MONITOR
      move.b  d5,d6             Ausgabeflag sichern
      clr     d5                auf Bildschirmausgabe
      bsr.s   AUSGABE           umschalten und Text
      move.b  d6,d5             ausgeben, dann Flag
```

```
ZURUECK1
      rts                         zurücksetzen
CRLF
      lea       CRLFTEXT,a6       CR und LF ausgeben
AUSGABE
      clr       d7                Zähler löschen
AUSGBELP
      tst.b     (a6)              prüfen, ob Ende erreicht
      beq.s     ZURUECK1          ja -> zurück
      addq      #1,d7             Zähler erhöhen
      move.b    (a6)+,d4          hole Zeichen und blende
      and       #$FF,d4           das obere Byte aus
AUSGABE1
      move      d4,-(sp)          ASCII-Code auf Stack
      tst.b     d5
      beq.s     MON
      move      #PRTOUT,-(sp)
      bra.s     AUSGABE2
MON
      move      #CONOUT,-(sp)     Befehl auf Stack legen
AUSGABE2
      trap      #GEMDOS           und GEMDOS aufrufen
      addq.l    #4,sp             Stack aufräumen
      tst.b     d5                Monitorausgabe ?
      beq.s     AUSGBELP          ja -> nächstes Zeichen
      tst.b     dØ                Druckerfehler
      bne.s     AUSGBELP          nein -> nächstes Zeichen
      pea       ERRTXT            Druckerfehlermeldung
      move      #PRTLINE,-(sp)    auf Bildschirm ausgeben
      trap      #GEMDOS           GEMDOS aufrufen
      addq.l    #6,sp             Stack aufräumen
PRINTERR
      move      #CONIN,-(sp)      Ein Zeichen mit GEMDOS-
      trap      #GEMDOS           Funktion CONIN einlesen
      addq.l    #2,sp             Stack aufräumen
      cmp.b     #13,dØ            RETURN-Taste ?
      beq.s     AUSGABE1          Wiederholung der Ausgabe
      cmp.b     #ESC,dØ           ESC-Taste ?
      bne.s     PRINTERR          Wiederholung der Abfrage
      bra       ENDE              Ende des Programms
          DATA

TITLE     dc.b    27,'E',27,'Y',33,62
          dc.b    'DIR-LISTER',1Ø,13,1Ø,13,Ø
PROMT     dc.b    1Ø,13,1Ø,13,'Laufwerk : ',Ø
CRLFTEXT  dc.b    1Ø,13,Ø
DISKNAME  dc.b    1Ø,13,1Ø,13,'Diskette: ',Ø
ERRTXT    dc.b    1Ø,13,1Ø,13,'Druckerfehler '
          dc.b    '[<ESC> = Prg.-Ende'
          dc.b    '  --  <CR> = Fortsetzung] :  ',Ø
DRUCKER   dc.b    1Ø,13,1Ø,13,'Ausgabegerät'
          dc.b    '  (Monitor = <Ø>'
          dc.b    '  -  Drucker = <1>  '
          dc.b    '-  Ende = <ESC>) : ',Ø
SPACE     dc.b    ' ',Ø
SUCHNAME  dc.b    'Suchfilename (RETURN = *.*) : ',Ø
TABELLE   dc.l    1ØØØØØØØØØ,1ØØØØØØØØ
          dc.l    1ØØØØØØØ,1ØØØØØØ,1ØØØØØ
```

```
            dc.l    10000,1000,100,10,1
ROTEXT      dc.b    ' * ',0
TASTE       dc.b    10,13,10,13,'Taste drücken',0
FILE        dc.b    '*.*',0
CURSON      dc.b    10,13,27,'e',0
CURSOFF     dc.b    10,13,27,'f',0

            BSS

DTABUFF     ds.b    46    DTABUFFER für GEMDOS
BUFFER      ds.b    250   enthält den aktuellen Pfad
DEZZAHL     ds.b    12    enthält umgewandelte Dez-Zahl
STACK       ds.b    1000  Stack
FILEN1      ds.b    2     Länge der Eingabe
FILEN       ds.b    15    Such-Filename

            END
```

Sortieren von ASCII-Files

Dieses kleine C-Programm sortiert Ihre abgespeicherten ASCII-Files. Der Befehl lautet dann

SORT QUELL_FILE ZIEL_FILE

Das Programm wurde mit dem Megamax, sowie Lattice C-Compiler getestet. Auch mit dem C-Compiler des Entwicklungssystems von Digital Research dürfte es keine Probleme geben.
Dass Programm sollte nur von Command aus gestartet werden.

```
/* ASCII - Datei Sortieren                6.11.86 / ekw          */
#define int short

#include "osbind.h"
#include "stdio.h"


main(argc,argv)
int argc;
char *argv[];
{
register int i,j,k;
FILE *fp;
register char *s;
char *fgets();
char buf[27000];
char *p[1200];
int scp();
int c;

        s=buf;

        if(argc != 3) {
                printf("\n Format: SORT  SOURCE_FILE  DESTINATION_FILE\n");
                Crawcin();
                exit(1);
```

ASCII - SORT

```
        }

        if((fp=fopen(argv[1],"r"))==NULL) {
                printf("File not found\n");
                Crawcin();
                exit(1);
        }

        k=0;
        while(fgets(s,82,fp) != NULL) {

                if(*s > 32) {
                        p[k++]=s;
                        while(*s++);
                }
        }

        fclose(fp);

        for(i=0;i<k-1;i++) {
                s=p[i];
                for(j=i+1;j<k;j++) {
                        if(c=scp(s,p[j])) {
                                p[i]=p[j];
                                p[j]=s;
                                s=p[i];
                        }
                }
                printf("%s",s);
        }

        if((fp=fopen(argv[2],"w"))==NULL) {
                printf("Write Error\n");
                Crawcin();
                exit(1);
        }

        for(i=0;i<k;i++) {
                fputs(p[i],fp);
        }
        putc(13,fp);
        putc(10,fp);

        fclose(fp);

        printf("\n ok. \n");
        Crawcin();
}


int scp(a,b)
register char *a, *b;
{
        while(*a) {
                if(*a > *b) return(1);
                if(*a++ < *b++) return(0);
        }
        return(0);
}
```

Slowdown

Ein Rechner kann normalerweise nicht schnell genug sein und alle Benutzer schreien nach noch schnelleren Computern. Aber hat es nicht schon jeder einmal erlebt, daß eine Bildschirmausgabe auf die Verarbeitungsgeschwindigkeit des menschlichen Geistes überhaupt keine Rücksicht nimmt? Oder der Highscore eines Computerspiels für den mit einer durchschnittlichen Reaktionszeit ausgestatteten normaler Anwender absolut unerreichbar bleibt? Könnte der Computer nicht ein wenig langsamer sein?

Er kann: Mit dem Programm „SLOWDOWN" läßt sich der ST bis auf ein Drittel seiner ursprünglichen Geschwindigkeit zurückdrosseln. Alles läuft praktisch nur noch in Zeitlupe ab.

WIE FUNKTIONIERT DAS PROGRAMM?

Jeder, der sich schon einmal näher mit Atari ST beschäftigt hat, weiß, daß in seinem Inneren der mächtige Prozessor 68000 von Motorola seinen Dienst verrichtet und den ST auch zu einer anständigen Arbeitsgeschwindigkeit verhilft. Dieser Prozessor besitzt nun neben vielen anderen schönen Eigenschaften auch die Möglichkeit, einzelne Befehle im sogenannten Trace-Betrieb abzuarbeiten. Dafür hat der 68000 in seinem Statusregister eine eigene Flag. Wird sie von einem Programm gesetzt, springt der Prozessor vor der Abarbeitung des nächsten Befehls über den Ausnahmevektor #9 in eine spezille Routine. Hier können nun das laufende Programm analysiert oder ähnliche Dinge im System verrichtet werden. Das Programm „SLOWDOWN" benutzt nun gerade diese Möglichkeit und „traced" praktisch jeden Befehl. Die Folge ist natürlich, daß das ganze System entsprechend verlangsamt wird.

Das Programm „SLOWDOWN" wird in den Rechner eingeladen und macht nun zunächst überhaupt nichts. Wird jetzt allerdings die Tastenkombination ALT+HELP, die normalerweise eine Hardcopy des Bildschirms veranlaßt, gedrückt, so wird jeder Prozessorbefehl getraced und der Rechner wird entsprechend gebremst. Ein erneuter Druck veranlaßt den ST, wieder seine normale Arbeitsgeschwindigkeit anzunehmen. Programme, bei denen keine Hardcopy möglich ist, lassen sich so leider auch nicht durch „SLOWDOWN" beirren.

Das Programm läuft sonst auf allen ST's – mit oder ohne ROM. Es ist vollständig in Assembler geschrieben. Wer keinen Assembler zur Verfügung hat, kann das GFA-Basiclisting eingeben. Es erzeugt eine lauffähige Datei direkt auf Diskette.

```
Assemblerlisting zu 'SLOWDOWN'
*****************************************
* SLOWDOWN.S              V.1.Ø         *
*---------------------------------------*
*                                       *
* ALT/HELP --> Slow - Fast              *
*****************************************
GEMDOS          =       1       ; Gemdosaufruf.
SUPER           =       $2Ø     ; Supervisor ein/aus.
KEEP            =       $31     ; Keep Process.
DUMPFLG         =       $4EE    ; Hardcopyflag.
*---------------------------------------*
MAIN:   BRA     INSTALL         ; Vektoren installieren.
*---------------------------------------*
NEWTRACE:
        TST     TFLG            ; Traceflag gesetzt ?
        BEQ     NO_MORE         ; nein.
        BSET    #7,(SP)         ; Tracebit setzen.
        ;       Hier ein paar NOP's und es wird noch
        ;       langsamer....
        ;       NOP
        ;       NOP
        ;
        RTE                     ;
NO_MORE:
        BCLR    #7,(SP)         ; Tracebit loeschen.
        RTE
*---------------------------------------*
VBLI:
        TST     DUMPFLG         ; ALT-HELP gedrueckt ?
        BNE     GOOLD           ; Nein.
        BSR     EFFECT          ; Bild invertieren.
                                ; (optional)
        BSET    #7,(SP)         ; Tracebit setzen.
        EOR     #-1,TFLG        ; Traceflag ein o. aus.
        MOVE    #-1,DUMPFLG     ; ALT-HELP Flag loeschen.
GOOLD:
        DC.W    $4EF9           ; Sprung zur alten VBL-
OLDVBL: DC.L    -1              ; Routine.
*---------------------------------------*
EFFECT:
        BSR     EFF             ; Routine invertiert kurz
EFF:                            ; den Screen.
        MOVEM.L DØ/AØ,-(SP)     ; Kann weggelassen werden
        MOVE.L  $44E,AØ         ; (siehe oben).
        MOVE    #8ØØØ-1,DØ
EFF_Ø1: EOR.L   #-1,(AØ)+
        DBRA    DØ,EFF_Ø1
        MOVEM.L (SP)+,DØ/AØ
        RTS
*---------------------------------------*
TFLG:   DC.W    Ø
*---------------------------------------*
INSTALL:
        CLR.L   -(SP)           ; Supermodus ein.
        MOVE    #SUPER,-(SP)
        TRAP    #GEMDOS
```

```
        LEA     6(SP),SP
        MOVE.L  D0,OLDSP          ; Stackpointer sichern.

        MOVE    SR,D0             ; Statusregister sichern.
        OR      #$700,SR          ; Interrupts aus.
        MOVE.L  #NEWTRACE,$24     ; Neuen Tracevektor eintragen.
        MOVE.L  $70,OLDVBL        ; Alten VBL-Vektor sichern.
        MOVE.L  #VBLI,$70         ; Neuen VBL-Vektor eintragen.
        MOVE    D0,SR             ; Interrupts ein.

        MOVE.L  OLDSP,-(SP)       ; Zurück in den User-Modus.
        MOVE    #SUPER,-(SP)
        TRAP    #GEMDOS
        LEA     6(SP),SP

        CLR.L   -(SP)             ; Programm verlassen.
        MOVE.L  #INSTALL-MAIN+$100,-(SP)
        MOVE    #KEEP,-(SP)       ; Speicherplatz nicht
        TRAP    #GEMDOS           ; freigeben.

*---------------------------------------*
.BSS
OLDSP:  DS.L    1
*---------------------------------------*

GFA-Basiclisting zu 'SLOWDOWN'.

' DATAGENERATOR V.1.0
Open "O",#1,"\SLOWDOWN.TOS"
Sum=0
Zeile=1
Do
  Read A$
  Exit If A$="-1"
  B$="&"+A$
  A=Val(B$)
  If A>=&H1000
    Print "DATAZEILE ";Zeile;
    If Sum=A-&H1000
      Print " OK"
    Else
      Print " FEHLER"
      Stop
    Endif
    Sum=0
    Inc Zeile

              A=0
            Else
              Out #1,A
            Endif
            Sum=Sum+A
          Loop
          Close #1
          Data 60,1A,00,00,00,B0,00,00,00,00,112A
          Data 00,00,00,04,00,00,00,00,00,00,1004
          Data 00,00,00,00,00,00,00,00,60,00,1060
          Data 00,5E,4A,79,00,00,00,5E,67,00,11E6
```

```
Data 00,08,08,D7,00,07,4E,73,08,97,124E
Data 00,07,4E,73,4A,78,04,EE,66,00,12E2
Data 00,18,61,00,00,1A,08,D7,00,07,1179
Data 0A,79,FF,FF,00,00,00,5E,31,FC,140C
Data FF,FF,04,EE,4E,F9,FF,FF,FF,FF,1833
Data 61,00,00,02,48,E7,80,80,20,78,132A
Data 04,4E,30,3C,1F,3F,0A,98,FF,FF,13BC
Data FF,FF,51,C8,FF,F8,4C,DF,01,01,163B
Data 4E,75,00,00,42,A7,3F,3C,00,20,1247
Data 4E,41,4F,EF,00,06,23,C0,00,00,12B6
Data 00,B0,40,C0,00,7C,07,00,21,FC,1350
Data 00,00,00,04,00,24,23,F8,00,70,11B3
Data 00,00,00,3A,21,FC,00,00,00,1A,1171
Data 00,70,46,C0,2F,39,00,00,00,B0,128E
Data 3F,3C,00,20,4E,41,4F,EF,00,06,126E
Data 42,A7,2F,3C,00,00,01,60,3F,3C,1230
Data 00,31,4E,41,00,00,00,06,28,40,112E
Data 0C,0A,06,0A,1026
Data -1
```

Superformat

Haben Sie sich nicht auch schon öfters geärgert, daß Sie nicht den gesamten ROM-Speicher Ihres Ataris auf einer einzigen Diskette unterbringen können, da das GEM Formatierungsprogramm „nur“ 720 KB schafft?

Damit kann ab jetzt Schluß sein, denn wenn Sie das nachfolgende Assemblerlistung in Ihren Computer eingeben, schafft Ihre Diskettenstation, vorausgesetzt Sie sind im Besitz einer SF 314, pro Diskette einen freien Speicherplatz von fast 930 KB.

Das Listing ist umfangreich dokumentiert, so daß Sie keine Probleme damit haben werden.

```
GI1             =       $FFFF8800       ;YM Register lesen+selekt
GI2             =       GI1+2           ;       schreiben
CMD             =       $80             ; Register WD 1772
TRK             =       $82             ;       "
SEKT            =       $84             ;       "
DAT             =       $86             ;       "
FLOCK           =       $43E            ; VBL Floppyflag
SECLEN          =   2                   ; Sektorlaenge = 512 Bytes
INTERL          =   6                   ; Interleave Faktor
SEPT            =   11                  ; Sektoren pro Track
MAXTRK          =   83                  ; Number of Tracks
CR              =   13                  ; Return..
LF              =   10                  ; LineFeed.
GEMDOS          =   1                   ; Gemdosaufruf.
XBIOS           =   14                  ; XBiosaufruf.
SUPER           =   $20                 ; Supervisor umschalten.
PRINT           =    $9                 ; Print   Line.
DIRCON          =   $7                  ; Direct CON in   ohne Echo.
PROTOBT         =   $12                 ; Bootsector generieren.
FLOPWR            =    $9               ; Sektoren schreiben.
*****************************************
```

```
START:
        CLR.L    -(SP)              ; Ganze Routinen im Supervisor.
        MOVE     #SUPER,-(SP)
        TRAP     #GEMDOS
        LEA      6(SP),SP
        MOVE.L   DØ,OLDSSP
AGAIN:
        LEA      IDENT,AØ           ; Ueberschrift ausgeben.
        BSR      PRTS
        BSR      INKEY              ; Auf Taste warten.
        CMP.B    #$1B,DØ            ; Falls   ESC dann Abbruch
        BEQ      EXIT               ;
        LEA      PRCLR,AØ           ;
        BSR      PRTS               ; Clear   Screen.
        MOVE.L   #$FFFF8606,A6      ; Floppyregister (indirekt).

        MOVE     #-1,FLOCK          ; Floppy VBI-Routine sperren.
        MOVE.B   #Ø,DØ              ; Laufwerk A
        MOVE.B   #Ø,D1              ; Seite Ø
        BSR      SELEKT
        BSR      RESTORE            ; Kopf auf Spur   Ø.

        CLR.B    SIDENO             ; Start   Ø
        CLR.B    TRKNO              ; Track   Ø

MLOOP:
        LEA      HEX,AØ             ; Tracknummer umrechnen
        MOVE.B   TRKNO,DØ           ; in Hex (2 Digits).
        LSR.B    #4,DØ              ;
        AND      #$F,DØ             ;
        MOVE.B   Ø(AØ,DØ),NUM       ;
        MOVE.B   TRKNO,DØ           ;
        AND      #$F,DØ             ;
        MOVE.B   Ø(AØ,DØ),NUM+1     ;
                                    ;
        LEA      STRING11,AØ        ; Und ausgeben...
        BSR      PRTS

        MOVE.B   #Ø,DØ              ; Laufwerk A
        MOVE.B   SIDENO,D1          ; Seite Ø
        BSR      SELEKT             ; Laufwerk selektieren.

        BSR      SETBUF             ; Puffer fuellen.
        BSR      WRTRK              ; Track formatieren.

        ADD.B    #1,SIDENO          ; Seite Ø
        CMP.B    #2,SIDENO          ; und auch fuer Seite 1.
        BNE      MLOOP              ;

        CLR.B    SIDENO             ; naechster Track.
        ADD.B    #1,TRKNO           ;
        CMP.B    #MAXTRK,TRKNO      ; letzter Track erreicht ?
        BEQ      CONT               ; ja.
        BSR      STEPIN             ; sonst formatieren...
        BRA      MLOOP              ;

CONT:
        BSR      RESTORE            ; Kopf auf Spur Ø.
        BSR      CLBU               ; Clear buffer.
```

```
            LEA     BOOT,A0          ; Bilde Bootsektor im
            LEA     BUFFER,A1        ; Buffer.
            MOVE    #40-1,D5
    COBT:
            MOVE.B  (A0)+,(A1)+
            DBRA    D5,COBT

            CLR     -(SP)            ; Protobt
            MOVE    #-1,-(SP)        ; schreibe neue Seriennr.
            MOVE.L  #$01000000,-(SP);
            PEA     BUFFER           ;
            MOVE    #PROTOBT,-(SP)   ;
            TRAP    #XBIOS           ;
            LEA     14(SP),SP        ;

            MOVE    #1,D6            ; Sektor 1
            CLR     D5               ; Seite 0
            BSR     WR_SEC           ; Schreibe Bootsektor.

            BSR     CLBU             ; Clear Buffer
            MOVE    #1,D5            ; Leeren Sektor
            BSR     WR_SEC           ; auf Seite 1 schreiben.

            MOVE    #2,D6            ; Leersektoren schreiben
    CLEAR:                           ; ab Sektor 2
            CLR     D5               ; Seite 0 und
            BSR     WR_SEC           ; Seite 1 fuer
            MOVE    #1,D5            ; FAT's ,etc...
            BSR     WR_SEC           ;
            ADD     #1,D6            ; Letzter Leersektor ?
            CMP     #12,D6           ;
            BNE     CLEAR            ; nein.

            BRA     AGAIN            ; Disk fertig....
    ****************************************
    EXIT:
            CLR     FLOCK            ; Floppy VBL freigeben.
            MOVE.L  OLDSSP,-(SP)     ; In Usermodus zurueck.
            MOVE    #SUPER,-(SP)
            TRAP    #GEMDOS
            LEA     6(SP),SP

            CLR     -(SP)            ; Zum Desktop....
            TRAP    #GEMDOS


****************************************
STEPIN:
        MOVE    #$50,D6          ; Stepin mit update...
        BRA     FLOPCMDS
****************************************
RESTORE:
        CLR     D6               ; Kopf auf Track 0
        CLR.B   TRKNO
****************************************
FLOPCMDS:                        ; CMD in D6 -- (D6=0:OK D6=-1:Fehler)
        MOVEM.L D5/D7,-(SP)
        AND.B   #$F0,D6          ;              (D0 enthaelt Status)
```

```
        OR.B    #$03,D6          ; 3 ms - no verify - Spin up Seq.
        MOVE.L  #$40000,D5       ; Timeout fuer WD-Cmd.
        MOVE    #CMD,(A6)        ; CMD Reg. selektieren.
        BSR     RDISKCT          ; Status lesen - Motor on ?
        BTST    #7,D0            ; Motor laeuft noch ?
        BNE     MTON             ; Ja.
        MOVE.L  #$60000,D5       ; Sonst Timout erhöhen.
MTON:   MOVE    D6,D7
        BSR     WDISKCT          ; Befehl ins CMD-Reg. schreiben.
CMWAIT: SUB.L   #1,D5            ; Timeout -1.
        BEQ     CMDERR           ; Cmd abbrechen.
        BTST    #5,$FFFFFA01     ; WD 1772 fertig ?
        BNE     CMWAIT
        BSR     RDISKCT          ; Status nach D0
        CLR     D6               ; Befehl erfolgreich...
CMDEX:  MOVEM.L (SP)+,D5/D7      ; zurueck.
        RTS

CMDERR:
        BSR.S   FORCEI           ; Befehl unterbrechen.
        MOVE    #-1,D6           ; Befehl nicht erfolgreich...
        BRA.S   CMDEX

****************************************
FORCEI:
        MOVE    #CMD,(A6)        ; Cmd-Register.
        MOVE    #$D0,D7          ; Force Interrupt.
        BSR     WDISKCT
        MOVE    #$F,D7           ; Warte einen Moment...
CMDWA:  DBRA    D7,CMDWA
        BSR     RDISKCT          ; Status lesen --> D0
        RTS

****************************************
WDISKCT:                         ; Schreibe WD
        BSR     WAIT1
        MOVE    D7,$FFFF8604     ; Schreibe WD-1772 Reg.
        BRA     WAIT1
RDISKCT:                         ; Lese WD-->D0
        BSR     WAIT1
        MOVE    $FFFF8604,D0
WAIT1:
        MOVE    D7,-(SP)         ; Warteschleife.
        MOVE    #$20,D7
WA01:   DBRA    D7,WA01
        MOVE    (SP)+,D7
        RTS

****************************************
SELEKT:
        ADD.B   #1,D0            ; D0 --> 0=Laufwk A
        LSL.B   #1,D0            ;        1=Laufwk B
        OR.B    D1,D0            ; D1 --> 0=Seite 0
        EOR.B   #7,D0            ;        1=Seite 1
        AND.B   #7,D0
        MOVE    SR,-(SP)         ; Interrupts sperren.
        OR      #$700,SR
```

```
        MOVE.B  #$E,GI1         ; Reg. sektieren.
        MOVE.B  GI1,D1          ; Reg. lesen.
        AND.B   #$F8,D1         ; Bits Ø-2 loeschen.
        OR.B    DØ,D1           ; Bits setzen.
        MOVE.B  D1,GI2          ; Reg. schreiben.
        MOVE    (SP)+,SR        ; Interrupt enable.
        RTS
****************************************
SETBUF:
        MOVE.B  #1,D3           ; Beginnen mit Sector 1.
        LEA     BUFFER,A2       ; Trackpuffer
        MOVE    #Ø4-1,D1        ; Trackvorspann mit (GAP I)
        MOVE.B  #$4E,DØ         ; Ø4 x $4e
        BSR     WBUFF
SEØ25:
        MOVE    D3,D4
SEØ2:
        MOVE    #Ø2-1,D1        ; Ø2 x $ØØ
        CLR.B   DØ
        BSR     WBUFF

        MOVE    #3-1,D1         ; 3 x $f5
        MOVE.B  #$F5,DØ         ; A1 in MFM
        BSR     WBUFF

        MOVE.B  #$FE,(A2)+      ; ID Address Mark
        MOVE.B  TRKNO,(A2)+     ; Tracknummer
        MOVE.B  SIDENO,(A2)+    ; Seite
        MOVE.B  D4,(A2)+        ; Sektornummer
        MOVE.B  #SECLEN,(A2)+   ; Sektorgroesse (2=512 Bytes)
        MOVE.B  #$F7,(A2)+      ; CRC Bytes schreiben

        MOVE    #22-1,D1        ; 22 x $4e
        MOVE.B  #$4E,DØ         ; ( GAP II)
        BSR     WBUFF

        MOVE    #12-1,D1        ; 12 x ØØ
        CLR.B   DØ
        BSR     WBUFF

        MOVE    #3-1,D1         ; 3 x $f5
        MOVE.B  #$F5,DØ         ; A1 in MFM
        BSR     WBUFF

        MOVE.B  #$FB,(A2)+      ; Data Adress Mark

        MOVE    #512-1,D1       ; 512 x virgin
        LEA     IDENT,AØ
SEØ1:
        MOVE.B  (AØ)+,(A2)+
        BNE.S   SEØ3
        LEA     IDENT,AØ
SEØ3:
        DBRA.S  D1,SEØ1

        MOVE.B  #$F7,(A2)+      ; CRC Bytes
```

```
        MOVE    #Ø2-1,D1        ; Ø2 x $4e
        MOVE.B  #$4E,DØ         ; ( GAP III )
        BSR     WBUFF

        ADD     #INTERL,D4      ; Interleave addieren.
        CMP.W   #SEPT,D4        ; Sektoren pro Track erreicht ?
        BLE     SEØ2            ; nein.
        ADD     #1,D3           ;
        CMP     #INTERL,D3
        BLE     SEØ25


        MOVE    #1ØØØ-1,D1      ; Sektorende fuellen.
        MOVE.B  #$4E,DØ

WBUFF:
        MOVE.B  DØ,(A2)+        ; Daten in Puffer schreiben
        DBRA.S  D1,WBUFF
SEEND:
        RTS
*****************************************
WRTRK:
        MOVE.L  #BUFFER,DØ
        MOVE.B  DØ,$FFFF86ØD    ; DMA-Low
        LSR.L   #8,DØ
        MOVE.B  DØ,$FFFF86ØB    ; DMA-Mid
        LSR.L   #8,DØ
        MOVE.B  DØ,$FFFF8609    ; DMA-High

        MOVE    #$19Ø,(A6)      ; DMA Status loeschen
        MOVE    #$9Ø,(A6)
        MOVE    #$19Ø,(A6)      ; DMA auf Write
        MOVE    #31,D7          ; DMA Sektor Count auf 31
        BSR     WDISKCT

        MOVE    #$18Ø,(A6)      ; 1772 selektieren.
        MOVE    #$FØ,D7         ; Write track...
        BSR     WDISKCT

        MOVE.L  #$4ØØØØ,D5
TEAGA:  BTST    #5,$FFFFFA01    ; 1772 fertig ?
        BEQ     MFPOK
        SUB.L   #1,D5
        BNE     TEAGA
        BSR     FORCEI

        LEA     FORMERR,AØ      ; Timer abgelaufen.
        BRA     PRTS

MFPOK:
        LEA     FORMOK,AØ       ; Track ok.
        BRA     PRTS
*****************************************
CLBU:
        MOVE    #127,D4         ; Clear Puffer.
        LEA     BUFFER,AØ
CLB1:
```

```
        CLR.L   (AØ)+
        DBRA.S  D4,CLB1
        RTS
*****************************************
WR_SEC:
        MOVE    #1,-(SP)        ; 1 Sektor schreiben.
        MOVE    D5,-(SP)        ; Seite
        CLR     -(SP)           ; Track #
        MOVE    D6,-(SP)        ; Sektor #
        MOVE    #Ø,-(SP)        ; Laufwerk A
        CLR.L   -(SP)
        PEA     BUFFER
        MOVE    #FLOPWR,-(SP)
        TRAP    #XBIOS
        LEA     2Ø(SP),SP
        RTS

*****************************************
PRTS:
        MOVE.L  AØ,-(SP)        ; Print Line.
        MOVE    #PRINT,-(SP)
        TRAP    #GEMDOS
        LEA     6(SP),SP
        RTS

*****************************************
INKEY:
        MOVE    #DIRCON,-(SP)   ; Wait for Key.
        TRAP    #GEMDOS
        ADDQ.L  #2,SP
        RTS


*****************************************
HEX:            DC.B    "Ø123456789ABCDEF"
PRCLR:          DC.B    $1B,"E",Ø
IDENT:          DC.B    $1B,"E",$1B,"f",$1B,"b1"
                DC.B    "*****************************************",CR,L
F
                DC.B    "*                                       *",CR,L
F
                DC.B    "*                                       *",CR,L
F
                DC.B    "*         SUPERFORMAT Vers. 1.Ø         *",CR,L
F
                DC.B    "*         ======================        *",CR,L
F
                DC.B    "*    Nur für doppelseitige Floppies !   *",CR,L
                DC.B    "*        923648 freie Bytes....         *",CR,L
F
                DC.B    "*****************************************",LF,C
R,LF
                DC.B    "    Bitte Disk in Laufwerk A einlegen",CR,LF
                DC.B    "    Taste druecken - ESC fuer Abbruch",CR,LF,Ø
STRING11:       DC.B    CR," Formatiere Track "
NUM:            DC.B    "ØØ.",Ø
FORMOK:         DC.B    " OK ",Ø
FORMERR:        DC.B    " FEHLER !",Ø
.EVEN
```

```
BOOT:           DC.W    $0,$0,$0,$0,$3DF8,$0300,$0202,$0100,$0270,$0022
                DC.W    $07F9,$0500,$0B00,$0200,$00,$00,$00,$00,$00,$00,
                $00
.EVEN
.BSS
OLDSSP:         DS.L    1       ; Alter SSP
TRKNO:          DS.B    1       ; Aktueller Track
SIDENO:         DS.B    1       ; Seite
SECNO:          DS.B    1       ; Aktueller Sector
.EVEN
BUFFER:
```

Optimiertes programieren in GFA-BASIC

Dieser Artikel soll Ihnen einige Tips und Tricks verraten, die Ihre BASIC-Programme beschleunigen. Wir verwenden dabei den GfA-BASIC-Interpreter. Die hier vermittelten Informationen gelten jedoch grundsätzlich für fast alle BASIC-Versionen, so daß Sie – auch wenn Sie in ST- oder Omikron-BASIC programmieren, diese Anregungen berücksichtigen sollten.

Wir stellen hier immer zwei fast identische Programme gegenüber, die den gleichen Zweck erfüllen, sich jedoch in der Ausführungsgeschwindigkeit stark unterscheiden. Die Leistungsdifferenzen geben wir in Prozent an, da die absoluten Zeitwerte uninteressant sind und von der jeweiligen Schleifenlänge abhängen.

Das erste Beispiel zeigt, daß man als Schleifenzähler einer FOR-NEXT-Schleife stets eine Integer-Variable verwenden sollte, sofern es die Problemstellung natürlich zuläßt. Das Programm 1b benötigt ganze 66 Prozent weniger Zeit als das fast identische Programm 1a:

Programm 1a: Zeit: 100%
For I=1 to 2000 Next I
Programm 1b: Zeit 34%
For I%=1 to 20000
Next I%

GfA-BASIC verfügt über die assemblerähnlichen Befehle INC und DEC zur Inkrementierung und Dekrementierung von Variablen. Der Befehl
INC A
entspricht der Anweisung
A=A+1

Vielen GfA-BASIC-Programmierern sind diese Befehle unbekannt, da sie zuvor mit anderen, nicht so leistungsfähigen Interpretern gearbeitet haben. Es lohnt sich jedoch, diese Möglichkeiten zu nutzen, da sie eine große Zeitersparnis bringen.

Programm 2a: Zeit: 100%
For I%=1 To 2000
A=A+I
Next I%
Programm 2b: Zeit: 62%
For I%=1 To 20000
Inc A
Next I%

Ganze 38 % Arbeitszeit erspart dieser einfache Befehl in diesem Fall. Das gleiche gilt natürlich auch für den DEC-Befehl.

Wenn man eine Variable mit einer Konstanten durch eine der vier Grundrechenarten verknüpfen will, bietet GfA-BASIC je zwei verschiedene Möglichkeiten, die wir hier gegenüberstellen wollen:

A=A+5	ADD A,5
A=A–5	SUB A,5
A=A*5	MUL A,5
A=A/5	DIV A,5

Die beiden folgenden Programmbeispiele, in denen die Multiplikation betrachtet wird, verdeutlichen, welche Methode die günstigere ist:

Programm 3a: Zeit 100%
For I%=1 To 20000
A=A*1
Next I%

Programm 3b: Zeit: 83%
For I%=1 To 20000
Mul A,1
Next I%

Programm 3b benötigt also 17% weniger Rechenzeit als die Version 3a. Man sollte also stets überlegen, ob man nicht die Befehle ADD, SUB, MUL oder DIV einsetzen kann.

Der letzte Tip beschäftigt sich mit dem Dreieckstausch. Dieser dient zum Austauschen von Variableninhalten. Wenn zum Beispiel die Werte von A und B vertauscht werden sollen, löst man dieses Problem häufig mit Hilfe der Hilfsvariablen C:

C=A
A=B
B=C

In GfA-BASIC gibt es jedoch den SWAP-Befehl, der dieses Verfahren überflüssig macht. Statt der drei Befehlszeilen verwendet man geschickterweise
SWAP A,B
wobei damit noch eine enorme Zeitersparnis verbunden ist. Rund 68% weniger Rechenzeit benötigt Programm 4b im Vergleich zu 4a:

Programm 4a: Zeit 100%
For I%=1 To 20000
C=A
A=B
B=C
Next I%

Programm 4b: Zeit: 32%
For I%=1 To 20000
Swap A,B
Next I%

Ein weiterer Vorteil liegt natürlich darin, daß keine Hilfsvariable zur Zwischenspeicherung benötigt wird.

Wir hoffen, daß Sie einige dieser Anregungen in eigenen Programmen verwenden können und diese damit schneller und leistungsfähiger werden.

Hacken ist in

Fast jeder Besitzer eines Homecomputers und dazu zählt letztendlich auch der Atari ST wegen seines erschwinglichen Preises, hat wohl schon einmal geträumt, mit einem „richtigen" Großcomputer zu kommunizieren. Der Traum kann sehr schnell real werden, glücklicherweise nicht so, wie es im Film *Wargames* dargestellt wurde, aber spannend wird es mit Sicherheit oft genug werden.
Die Hackerei, wie sie auf neudeutsch bezeichnet wird, läßt sich wesentlich treffender mit Datenfernübertragung beschreiben, und diese Bezeichnung klingt schon harmloser, wenngleich sich an der Sache nichts ändert. Sie bleibt auch völlig legal, niemand braucht Angst vor dem Staatsanwalt zu haben, solange er nur in den Mailboxen herumstöbert.
Doch zuerst einmal, was benötigt der Einsteiger, um mit anderen Rechnern kommunizieren zu können? Mit der Voraussetzung, daß Sie bereits Besitzer eines Atari ST sind oder vorhaben, sich demnächst einen zu kaufen, haben Sie schon einmal den Grundstein. Was nun nur noch benötigt wird, sind ein Akustikkoppler oder ein Modem und ein Terminalprogramm. Mit „Mini-Term" stellen wir Ihnen ein solches Programm vor. Kaum erwähnenswert erscheint die Tatsache, daß noch ein Telefonapparat zur Verfügung stehen muß. □

```
        /* Terminalprogramm */

/*      Verwendete Systemaufrufe:
Die  Aufrufe  sind    in  osbind.h   definiert.
Sie sind alle vom Typ long(32Bit). Bei Zuweisung
auf ein short (16 Bit)  findet eine automatische
Typconversion  statt.
Statusabfragen: Test, ob Device bereit ist, Daten zu
uebernehmen  bzw.  zu  liefern  (Ø = nein,  ja = -1).

       Cconis()         Terminal Eingabe
       Cconos()         Terminal Ausgabe
       Cprnos()         Drucker aus
       Cauxis()         V24 ein
       Causos()         V24 aus

Zeichen-Ausgabe:
       Cconout(c)       Bildschirm
       Cprnout(c)       Drucker
       Cauxout(c)       V24
Zeichen-Eingabe:
 Cconin()          Tastatur, longword: low word: ascii, Ø
                             bei Sondertasten.
                             high word:
                             Tastennummer (59-68) bei
                             Funktionstasten F1 - F1Ø
                             Aufruf in inc()
       Cauxin()          V24
*/
#include "osbind.h"
#include "stdio.h"
#define END 26              /* End of File = CRTL-Z */
short prflag=1;             /* Drucker ein/aus 1/Ø  */
short lfflag=1;             /* Auto-LF ein/aus      */
short ecflag=1;             /* Local Echo ein/aus   */
char name[1ØØ]="test.t";
main()
{
register short c, ic,oc,iv,ov;
       Cconout(27);
       Cconout('E');        /* clear screen */
       info();
       pronoff();
       autolf();
       echo();
       iv=ic=ov=oc=Ø;
       while(1) {
        c=inc();
         if(c<Ø) {
              if(c== -1Ø) break;
         switch(c) {      /* Funktionstastenverteiler */
                  case -1:
                      pronoff();
                      break;
                  case -2:
                      send();
                      break;
                  case -3:
```

```
                        get();
                        break;
                    case -4:
                        autolf();
                        break;
                    case -5:
                        echo();
                        break;
                    default :
                        info();
                        break;
                }
                continue;
            }
            if(!iv) iv=inv24();
                if(!oc) {
                        oc=iv;
                        iv=0;
                }
                if(oc) oc=out(oc);
                if(!ic) ic=c;
                if(!ov) {
                        ov=ic;
                        ic=0;
                }
                if(ov) ov=outv24(ov);
        }
}
stcopy(a,b)             /* String kopieren a -> b */
register char *a, *b;
{
        while(*b++ = *a++);
}
pronoff()                       /* Drucker ein/aus  */
{
        prflag= !prflag;
        pr(" ");
        p0("Drucker ");
        if(prflag) pr("ein.");
        else pr("aus.");
}
autolf()                        /* Auto-LF      */
{
static char *text[2]= {
        "Auto - Line Feed aus.",
        "Auto - Line Feed ein."
        };

        lfflag= !lfflag;
        pr(" ");
        pr(text[lfflag]);
}
echo()                          /* Local Echo   */
{
static char *text[2]= {
        "Lokales Echo aus.",
        "Lokales Echo ein."
        };
        ecflag= !ecflag;
```

```
        pr(" ");
        pr(text[ecflag]); }


send()                              /* Datei senden */
{
FILE *fp, *fopen();
register short c,x,y,ac;

        pØ("Datei senden. ");
        if(filename()) return(Ø);
        ac=x=y=c=Ø;

        if((fp=fopen(name,"r"))!= (FILE *)Ø) {
         while(inc()!= -1Ø) {
                 x=inv24();
                 if(c) c=outv24(ac=c);
                 else {    /* LF --> CR+LF  */
                         if(ac == 13) c=1Ø;
                         else {
                         if((c=getc(fp))== EOF) break;
                            else if(c==1Ø) c=13;
                         }
                 }
                        if(y) y=out(y);
                        else y=x;
                 }
          while(outv24(END)) if(inc()== -1Ø) break;
                 fclose(fp);
                 pr("Uebertragung beendet");
        } else pr("geht nicht");
}
get()               /* Datei einlesen  */
{
FILE *fp, *fopen();
register short c,x,y,p;
        pr("");
        pØ("Datei empfangen. ");
        if(filename()) return(Ø);
        p=x=y=c=Ø;
        if((fp=fopen(name,"w"))!= (FILE *)Ø) {
           while((x=inc())!= -1Ø) {
            if(c) {
             if(!p || c!=1Ø) putc(c,fp);
              if( p=((c==13) && (lfflag)) ) putc(1Ø,fp);
                 while(out(c)) if(inc() == -1Ø) break;
                         c=Ø;
                    } else if((c=inv24())==END) break;
                     if(x>Ø && y==Ø) y=x;
                     else if(x == -1) pronoff();
                        if(y) y=outv24(y);
                }
                fclose(fp);
                pr("Uebertragung beendet");
        } else pr("geht nicht");
}
filename()    /* Dateinamen eingeben */
{
char x[1ØØ];
```

```
        pr(name);
        stcopy(name,x);
        pØ("neuer Name:");
        input(x);
        if(x[Ø]=='\Ø' || x[Ø]==' ') return(1);
        stcopy(x,name);
        return(Ø);
}
info()          /* F-Tasten-Info */
{
  pr("");
  pr("F1 : Drucker ein/aus");
  pr("F2 : Datei senden");
  pr("F3 : Datei empfangen");
  pr("     Dateiname: RETURN -> alten Namen uebernehmen");
  pr("     SPACE + RETURN -> Abbruch");
  pr("F4 : Auto-Linefeed ein/aus");
  pr("F5 : Lokales Echo ein/aus");
  pr("F1Ø: Abbruch");
  pr("");
input(s)        /* Stringeingabe */
char *s;
{
register short c,ecx;
short z=Ø;
        ecx=ecflag;
        ecflag=Ø;
        while((c=waitin())!=13) {
           if(c<=Ø) continue;
           if(c==8) {       /* backspace     */
                    if(z) {
                             z--;
                             *(--s)=' ';
                             Cconout(c);
                             Cconout(' ');
                    }
           } else {
                    z++;
                    *s++ =c;
           }
           Cconout(c);
  }
        inc();  /* LF ueberlesen    */
        pr("");

        if(z) *s = '\Ø';
        ecflag=ecx;
        return(c);
}
pØ(s)  /* Stringausgabe  */
char *s;
{
register short c;
        while(c= *s++) Cconout(c);
}
pr(s)
char *s;
{
        pØ(s);
```

```
        Cconout(13);
        Cconout(10);
}


waitin()  /* Tastatureingabe mit warten  */
{
register short c;
        while(!(c=inc()));
        return(c);
}

inc()             /* Tastatureingabe                    */
{                 /* Return=Zeichen; 0: keine Eingabe */
register long c,x;    /* -1 ... -10: F1 ... f10      */
static short crflag=0;

  if(crflag) {     /* Letztes Zeichen CR: Return LF */
                crflag=0;
                if(ecflag) out(10);
                return(10);
        }
        if(!Cconis()) return(0);  /* keine Eingabe  */
        c=Crawcin();              /* Zeichen holen  */
        x= c/655361;
        c%= 655361;
        if(c) {                       /* ASCII - Taste   */
                if(lfflag && c==131) crflag=1;
                if(ecflag) out((short)c);
                return((short)c);
        }
                                              /* Sondertaste */
        if(x<59 || x>68) return(0);
        return(58-(short)x);              /* Funktionstasten*/
}
inv24()      /* Zeichen von v24 holen */
{
register short c;
        if(!Cauxis()) return(0);
        c= Cauxin();
        c &= 255;
        return(c);
}
outv24(c)
register short c;
{
        if(!Cauxos()) return(c);
        Cauxout(c);
        return(0);
}
out(c)     /* Zeichen auf BS und Drucker ausgeben */
register short c;
{
static short flag=0;
       if(Cconos() && (!prflag || Cprnos()) ) {
                if(flag) {
                    flag=0;
                    if(c==10) return(0); }
```

```
                if(lfflag && c==13) flag=1;
                Cconout(c);
                if(prflag) Cprnout(c);
                if(flag) {
                        Cconout(10);
                        if(prflag) Cprnout(10);
                }
                return(0);
        } else return(c);
}
```

17 & 4

Unser Kartenspiel 17 & 4 läuft ausschließlich auf einem Atari ST in Verbindung mit einem Monochrom-Monitor. Am Anfang jedes Spiels wird bei der Eröffnung um den Einsatz gebeten. Ein Druck auf die linke Maustaste erhöht den Einsatz des Spielers um jeweils 5 Mark. Die rechte Maustaste setzt den Einsatz um 5 Mark herunter. Durch gleichzeitiges Betätigen beider Maustasten wird das Spiel dann gestartet. Möchte der Kartenspieler eine neue Karte, so erreicht er dies wiederum durch einen Klick mit der linken Maustaste. Mit der rechten Maustaste wird die Kartenausgabe gestoppt. Erhält der Spieler vom Rechner zwei Asse, so hat er das Spiel gewonnen und der Einsatz wird seinem Spielkonto gutgeschrieben. Gespielt wird mit einem 32er-Blatt, das nach jedem vierten Spiel neu gemischt wird. □

```
        ######################################
        #                                    #
        #       Programm: 17 und 4           #
        #                                    #
        #       Autor:    Cord Ahrens        #
        #                                    #
        #           (c) by C A V             #
        #                                    #
        ######################################
'
'
DIM spk%(32),ord%(32)
kasse1=200
kasse2=50
HIDEM
COLOR 1
PBOX 0,0,640,400
@titel
PBOX 0,0,640,400
spiel:
FOR i=1 TO 32
  ord%(i)=RANDOM(20)+1
NEXT i
z=0
REPEAT
  ADD z,1
  FOR i=32 TO 1 STEP -1
    IF ord%(i)=z THEN
      ADD aa,1
      spk%(aa)=i
    ENDIF
  NEXT i
```

```
UNTIL z=21
aa=0
misch=1
durchlauf:
DEFFILL 0,0
PBOX 13,10,627,37
PRINT AT(5,2);"Computer...Punkte.:.............Konto.:.
                ";kasse1;"........................."
PBOX 13,202,627,229
PRINT AT(5,14);"Spieler....Punkte.:.............Konto.:.
                 ";kasse2;".........Einsatz.:........."
IF misch=1
  PRINT AT(63,2);"Neu gemischt"
  misch=0
ENDIF
PAUSE 5
DEFFILL 1,0,0
PRBOX 15,40,115,180
DEFFILL ,2,21
PBOX 25,50,105,170
PAUSE 15
DEFFILL ,0,0
PRBOX 95,44,195,184
DEFFILL ,2,21
PBOX 105,54,185,174
betrag=0
PRINT AT(71,14);betrag
REPEAT
  IF MOUSEK=1
    betrag=betrag+5
  ENDIF
  IF MOUSEK=2
    betrag=betrag-5
  ENDIF
  IF betrag<0
    betrag=0
  ENDIF
  IF betrag>kasse2
    betrag=kasse2
  ENDIF
  PRINT AT(71,14);betrag;"."
  PAUSE 5
UNTIL MOUSEK=3
x=65
y=302
GOSUB geber
spieler=wert
x=145
y=305
GOSUB geber
wert=wert+spieler
PRINT AT(25,14);wert
spieler=wert
IF spieler=22
  spitze=1
  GOTO spitze
ENDIF
DO
```


IMPRESSUM

ST-LISTING erscheint in der CA-Verlags GmbH, Heßstraße 90, D-8000 München 40, Tel.: 089/1298011, Telex: 5214428 cav d

VERANTWORTLICH FÜR DEN INHALT: David Krasucki

STÄNDIGE MITARBEITER: Thomas Bosch, Jochen Rohr, Torsten Seibt, Gert Seidel, Oliver Steinmeier, Volkard Werner, Klaus Wahlers.

GESCHÄFTSFÜHRER: Werner E. Seibt

ANSCHRIFT FÜR ALLE VERANTWORTLICHEN: Postfach 1161, 8044 Unterschleißheim, Tel.: 089/1298011, Telex: 5214428 cav-d

Es gilt Preisliste Nr. 9 vom 1.6.1988. Media-Unterlagen bitte anfordern.

VERTRIEB: Verlagsunion Wiesbaden

Printed in Germany by ADV, Aindlingerstraße 17–19, 8900 Augsburg 1

```
    IF MOUSEK=1
      x=x+80
      y=y+3
      GOSUB geber
      wert=wert+spieler
      PRINT AT(25,14);wert
      spieler=wert
    ENDIF
    EXIT IF MOUSEK=2
    EXIT IF spieler=>20
  LOOP
  x=65
  y=110
  GOSUB geber
  comp=wert
  x=145
  y=113
  GOSUB geber
  wert=wert+comp
  PRINT AT(25,2);wert
  comp=wert
  DO
    EXIT IF spieler>21 OR comp=>20 OR comp=>spieler
    x=x+80
    y=y+3
    GOSUB geber
    wert=wert+comp
    PRINT AT(25,2);wert
    comp=wert
  LOOP
  spitze:
  IF spitze=0
    IF (spieler<22 AND spieler>comp)
                   OR (spieler<22 AND comp>21)
      PRINT AT(30,14);"Sieger"
      konto=2
    ELSE
      PRINT AT(30,2);"Danke"
      konto=1
    ENDIF
  ELSE
    PRINT AT(30,14);"Sieger"
    konto=2
    spitze=0
  ENDIF
  @kasse
  wert=0
  comp=0
  spieler=0
  i=36
  REPEAT
    ADD i,1
    COLOR 1
    LINE 2,i,635,i
    LINE 2,i+191,635,i+191
  UNTIL i=203
  COLOR 1
  durchlauf=durchlauf+1
```

```
IF durchlauf=4
  aa=0
  durchlauf=0
  GOTO spiel
ENDIF
GOTO durchlauf
PROCEDURE kasse
  IF konto=1 THEN
    kasse1=kasse1+betrag
    kasse2=kasse2-betrag
  ELSE
    kasse1=kasse1-betrag
    kasse2=kasse2+betrag
  ENDIF
  REPEAT
  UNTIL MOUSEK<>0
  IF kasse2<1 THEN
    GOSUB verloren
  ENDIF
  IF kasse1<=0 THEN
    GOSUB kasse_alle
  ENDIF
RETURN
PROCEDURE geber
  COLOR 1
  aa=aa+1
  spk=spk%(aa)
  IF spk=1 OR spk=9 OR spk=17 OR spk=25
    wert=7
    @kar
    a=x
    b=y-20
    @far
    FOR i=0 TO 40 STEP 40
      FOR w=10 TO 90 STEP 40
        a=x-20+i
        b=y+50-w
        @far
      NEXT w
    NEXT i
  ENDIF
  IF spk=2 OR spk=10 OR spk=18 OR spk=26
    wert=8
    @kar
    a=x
    FOR w=0 TO 56 STEP 56
      b=y-28+w
      @far
    NEXT w
    FOR i=0 TO 40 STEP 40
      FOR w=0 TO 100 STEP 50
        a=x-20+i
        b=y+50-w
        @far
      NEXT w
    NEXT i
  ENDIF
  IF spk=3 OR spk=11 OR spk=19 OR spk=27
```

```
    wert=9
    @kar
    a=x
    b=y
    @far
    FOR i=0 TO 40 STEP 40
      FOR w=-2 TO 100 STEP 32
        a=x-20+i
        b=y+46-w
        @far
      NEXT w
    NEXT i
  ENDIF
  IF spk=4 OR spk=12 OR spk=20 OR spk=28
    wert=10
    @kar
    a=x
    FOR w=0 TO 60 STEP 60
      b=y-30+w
      @far
    NEXT w
    FOR i=0 TO 40 STEP 40
      FOR w=0 TO 90 STEP 30
        a=x-20+i
        b=y+46-w
        @far
      NEXT w
    NEXT i
  ENDIF
  IF spk=5 OR spk=13 OR spk=21 OR spk=29
    wert=2
    @kar
    a=x
    b=y+44
    @far
    x1=10
    y1=-10
    RESTORE bube
    FOR i=1 TO 32
      READ x2,y2
      DRAW x+x1,y+y1 TO x+x2,y+y2
      x1=x2
      y1=y2
    NEXT i
    LINE x-10,y-12,x+10,y-12
    LINE x-10,y-10,x+10,y-10
    CIRCLE x,y-50,9
    FILL x,y-8
  ENDIF
  IF spk=6 OR spk=14 OR spk=22 OR spk=30
    wert=3
    @kar
    a=x
    b=y+40
    @far
    x1=10
    y1=-14
    RESTORE dame
```

```
    FOR i=1 TO 20
      READ x2,y2
      DRAW x+x1,y+y1 TO x+x2,y+y2
      x1=x2
      y1=y2
    NEXT i
    CIRCLE x,y-48,6
    LINE x-4,y+26,x-8,y+26
    LINE x+4,y+26,x+8,y+26
    LINE x+8,y-18,x-8,y-18
  ENDIF
  IF spk=7 OR spk=15 OR spk=23 OR spk=31
    wert=4
    @kar
    a=x
    b=y+44
    @far
    RESTORE koenig
    FOR i=0 TO 23
      READ x1,y1,x2,y2
      LINE x+x1,y+y1,x+x2,y+y2
    NEXT i
    DRAW x-8,y-58 TO x,y-60 TO x+8,y-58
    CIRCLE x+12,y-24,4
    CIRCLE x,y-46,8
  ENDIF
  IF spk=8 OR spk=16 OR spk=24 OR spk=32
    wert=11
    @kar
    IF spk=8 THEN
      DEFTEXT 1,0,0,4
      TEXT x-12,y-24,"Cord"
      TEXT x-18,y-17,"Ahrens"
    ENDIF
    a=x
    b=y
    @far
    PAUSE 5
  ENDIF
RETURN
PROCEDURE far
  IF spk=>1 AND spk<=8
    PCIRCLE a-5,b,4
    PCIRCLE a+5,b,4
    PCIRCLE a,b-6,4
    LINE a,b-6,a,b+8
  ENDIF
  IF spk=>9 AND spk<=16
    PCIRCLE a-5,b,4
    PCIRCLE a+5,b,4
    LINE a-5,b,a+5,b
    LINE a,b,a,b+8
    LINE a-8,b,a,b-10
    LINE a+8,b,a,b-10
    COLOR 1
    FILL a,b-4
  ENDIF
  IF spk=>17 AND spk<=24
```

```
    COLOR 1
    DEFFILL 2,2
    CIRCLE a-5,b,4,0,2300
    CIRCLE a+5,b,4,3100,1800
    LINE a-2,b,a+2,b
    LINE a-8,b+2,a,b+10
    LINE a+8,b+2,a,b+10
    FILL a,b+4
  ENDIF
  IF spk=>25 AND spk<=32
    COLOR 1
    DRAW a-8,b TO a,b-8 TO a+8,b TO a,b+8 TO a-8,b
    DEFFILL 2,2
    FILL a,b
  ENDIF
RETURN
PROCEDURE kar
  RESTORE kart
  FOR i=2 TO wert
    READ k$
  NEXT i
  kart:
  DATA B,D,K,0,0,7,8,9,10,A
  DEFFILL ,0,0
  PRBOX x-50,y-70,x+50,y+70
  DEFTEXT 1,0,0,13
  DEFFILL 1,1,4
  TEXT x-44,y-50,14,k$
  TEXT x+33,y-50,14,k$
  DEFTEXT 1,0,1800,13
  TEXT x-33,y+50,14,k$
  TEXT x+44,y+50,14,k$
RETURN
PROCEDURE verloren
  CLS
  DEFTEXT ,0,0,32
  TEXT 190,140,"Tja - verloren !"
  TEXT 110,200,"Wenn Sie wieder Geld haben,"
  TEXT 180,250,"besuchen Sie uns."
  i=450
  DO
    i=i-1
    TEXT 180,i,"Auf Wiedersehen !"
    EXIT IF i=300
    EXIT IF MOUSEK<>0
  LOOP
  m$="Na,zu Geld gekommen?"
  ALERT 2,m$,1,"Ja|Nein",b
  IF b=2
    CLS
    SETCOLOR 0,1
    END
  ENDIF
  RUN
RETURN
PROCEDURE kasse_alle
  i=0
  CLS
```

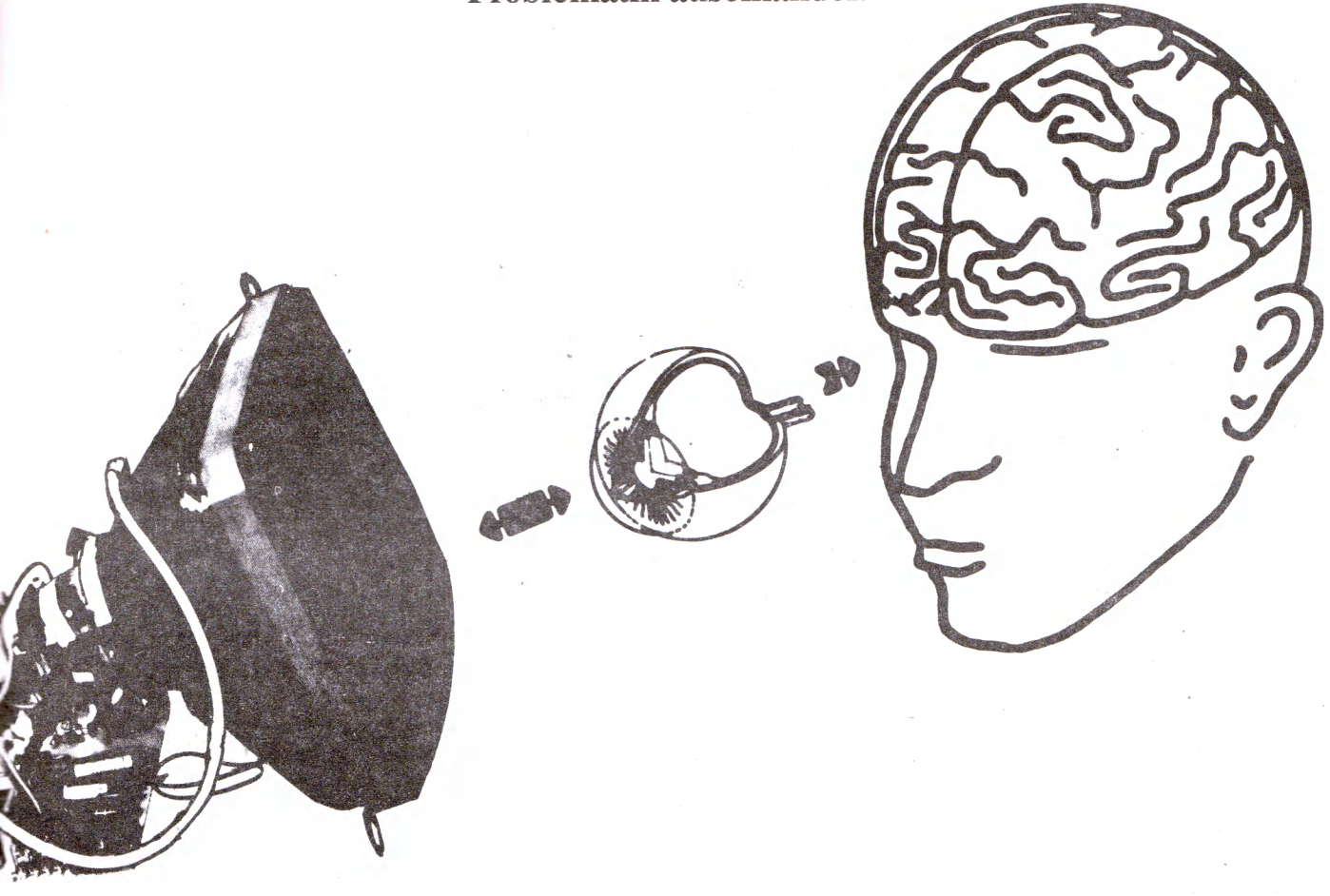
```
  GRAPHMODE 3
  REPEAT
    BOX i MOD 640,i MOD 400,639-i MOD 640,399-i MOD 400
    ADD i,8
  UNTIL i>=6400
  GRAPHMODE 1
  DEFTEXT ,0,0,32
  TEXT 60,145,520,"Sie haben die Bank gesprengt."
  TEXT 195,180,"Glückwunsch !!!"
  PAUSE 150
  m$="Ihr Nachbar hat verloren.|Die Bank hat wieder Geld!"
  ALERT 0,m$,1,"Noch Mal|Ich gehe",b
  IF b=2
    CLS
    SETCOLOR 0,1
    END
  ENDIF
  RUN
RETURN
PROCEDURE titel
  DEFTEXT 1,0,0,32
  TEXT 245,180,150,"Siebzehn"
  TEXT 315,220,"&"
  TEXT 290,260,"Vier"
  DEFTEXT ,0,0,6
  TEXT 245,278,150,"Mai,1988"
  COLOR 0
  FOR i=5 TO 250 STEP 4
    BOX 230-i*1.6,140-i,410+i*1.6,280+i
  NEXT i
  PAUSE 160
  COLOR 0
RETURN
'
bube:
'
DATA 18,-16,12,-28,10,-12,10,-4,14,16,14,30,
     20,30,20,34,8,34,6,16,0,-2,-6,16
DATA -8,34,-20,34,-20,30,-14,30,-14,16,-10,
     -4,-10,-12,-12,-28,-18,-16,-10
DATA -10,-10,-4,-26,-16,-16,-36,-4,-38,-4,-42,
     4,-42,4,-38,16,-36,26,-16,10,-4
dame:

DATA 10,-14,12,-22,8,-34,8,-18,20,24,-20,24,-8,-18,
     -8,-34,-12,-22,-10,-14
DATA -11,-10,-16,-20,-12,-38,-2,-40,-1,-42,1,-42,2,
     -40,12,-38,16,-20,11,-10
'
koenig:
'
DATA -20,32,20,32,20,32,20,-12,20,-12,16,-34,16,-34,
     4,-36,4,-36,4,-38,-20,32,-20,-12
DATA -20,-12,-16,-34,-16,-34,-4,-36,-4,-36,-4,-38,-4,
     -36,4,-28,4,-28,4,32,-7,-36,0,-24
DATA 0,-24,0,32,4,-36,0,-32,6,-36,2,-30,4,-16,12,-24,4,
     -18,11,-22,4,-14,11,-22,0,-54,0,-60,-6,-52,-8,-58,
      6,-52,8,-58,-4,34,-8,34,4,34,8,34,20,30,-20,30
```

Neue Monitortechnologie

Auge in Auge

Gewerkschaften und Betriebsärzte streiten sich schon lange mit der Arbeitgeberseite über das Für und Wider von Bildschirmarbeitsplätzen. Sie seien gesundheitsschädlich und nicht zumutbar, argumentieren die einen. Alternativen sind nach Ansicht der Arbeitgeber nicht abzusehen. Auch die Hersteller von Monitoren setzen sich mit der Problematik auseinander.



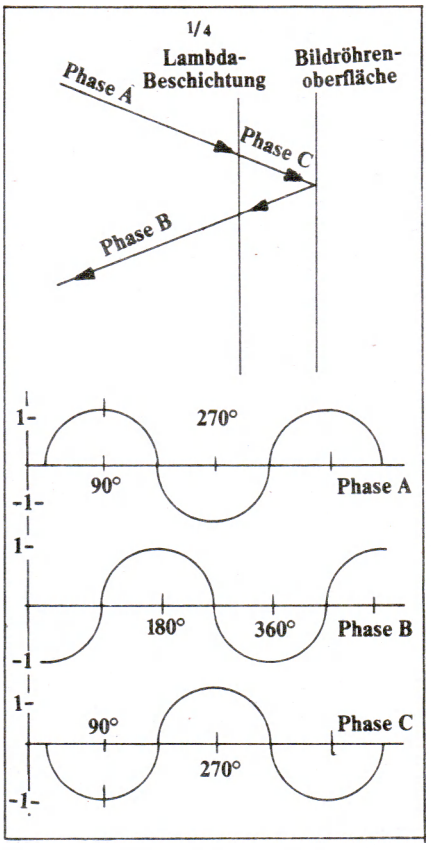
Die meisten Firmen, die derzeit versuchen, sich im Markt der hochwertigen Peripheriegeräte zu etablieren, orientieren sich an bereits existierenden Produkten der Konkurrenz.
Anders die Marketing-Linie der belgischen Firma Etap. Ihre Devise heißt: Erst Grundlagenforschung betreiben und dann produzieren. Resultat ist eine Reihe von Erkenntnissen, die jeden, der an einem Bildschirm arbeitet oder solche Arbeitsplätze plant, interessieren sollten.
Personen, die täglich vier bis fünf Stunden, an 200 Arbeitstagen, und das 30 Jahre lang an einem Bildschirm arbeiten, verbringen, hochgerechnet 24.000 Stunden ihres Arbeitslebens vor ihrem Monitor. Wenn es durch Verbesserung der Arbeitsbedingungen gelingen sollte, eine Steigerung der Produktivität dieser Personen von zehn Prozent zu erzielen, so ist dies ein Fortschritt.
Die Ergebnisse der Grundlagenforschung von Etap zeigen eine denkbare Steigerung der reinen Arbeitsgeschwindigkeit des Benutzers um 25 Prozent. Hierzu kommen noch derzeit schlecht meßbare Verbesserungen bei der Verhütung von Augenschäden und die Steigerung des allgemeinen Wohlbefindens von Personen, die am Bildschirm arbeiten müssen.

Das Auge im Detail

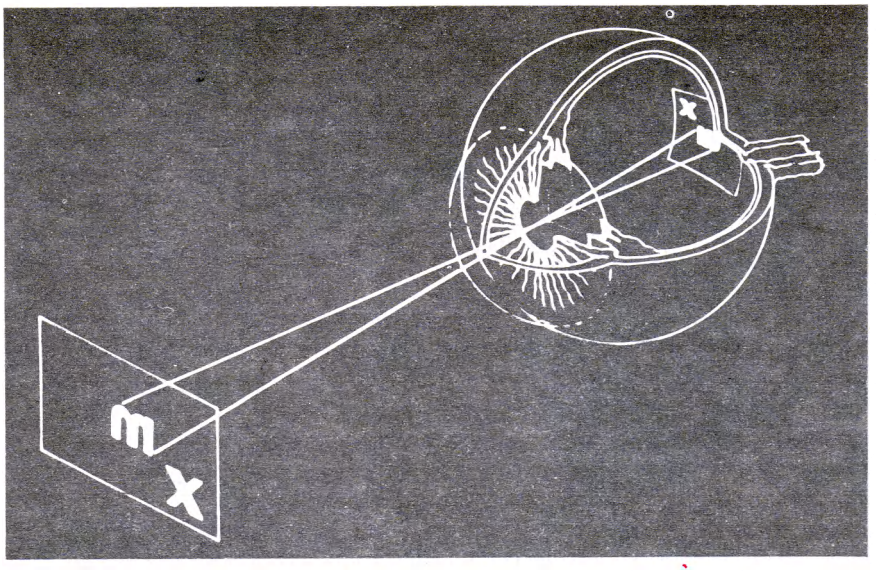
Der Monitor eines Computers und das menschliche Auge sind zwei Komponenten, die auf einander abgestimmt werden müssen. Dies ist nur möglich, wenn beider Aufbau und Funktion bis ins Detail bekannt ist.
Die wichtigsten Teile des menschlichen Auges sind die Linse, die Pupille und die Netzhaut (Retina). Die Linse projiziert eine Abbildung auf die Netzhaut und sorgt für die Schärfe dieser Abbildung. Da unsere Augen es gewohnt sind, immer scharf zu sehen, versuchen unsere Augen beim Lesen einer unscharfen Vorlage ständig durch Strecken und Dehnen von Muskeln die Linse so einzustellen, daß eine scharfe Abbildung auf der Netzhaut entsteht. Dies ist bei Verwendung einer unscharfen Vorlage ein hoffnungsloses Unterfangen. Unsere Augen ermüden durch das ständige Nachregulieren sehr schnell. Wird trotzdem weiter gearbeitet, läßt die Konzentration nach, und die Fehlerhäufigkeit des Arbeitenden steigt stark an. Andere Beschwerden, wie zum Beispiel Kopfschmerzen, folgen.
Jeder, der sich schon einmal mit Fotografie beschäftigt hat, kennt den Begriff Schärfentiefe. Gemeint ist eine Abhängigkeit zwischen Blendenöffnung und der Tiefe des Bereichs, der scharf dargestellt werden kann.
Die Pupille übernimmt im menschlichen Auge die Funktion der verstellbaren Blende eines Foto-Objektives. Je nach Lichteinfall öffnet oder schließt sich die Pupille. Bezogen auf die Arbeit am Bildschirm heißt dies: Ist der Bildschirm sehr hell, schließt sich die Pupille und eine große Schärfentiefe wird erreicht. Ist die Bildschirmdarstellung dunkel, so öffnet sich die Pupille und das Auge muß die Sehschärfe viel häufiger regulieren.

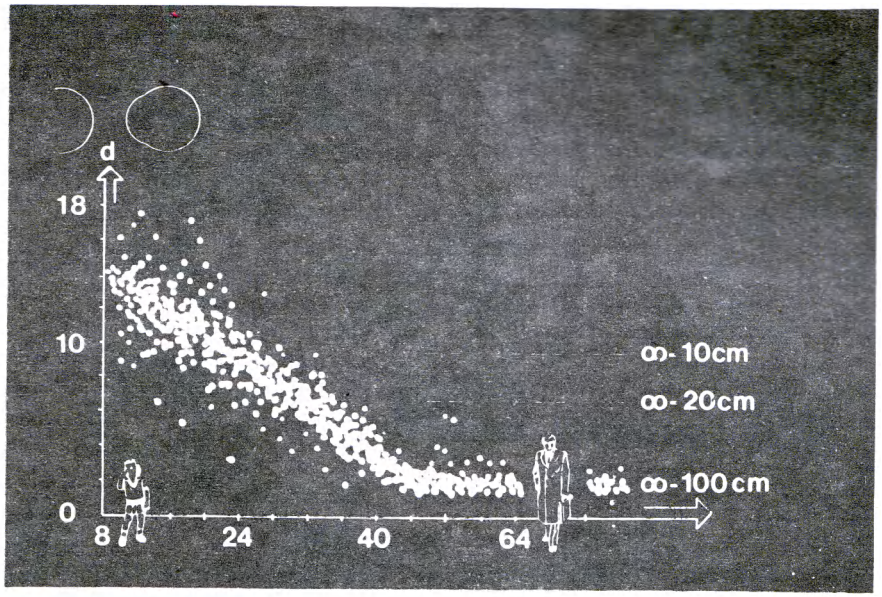
Ältere Augen brauchen mehr Licht

Personen, die ständig zwischen Papiervorlagen und Monitordarstellung wechseln, strapazieren bei großen Helligkeitsunterschieden ihre Augen erheblich mehr, als bei nahezu gleicher Helligkeit von Papier- und Bildschirmdarstellung. Ein weiterer Faktor ist die mit steigendem Alter abnehmende Empfindlichkeit unserer Augen. Je älter die Person ist, die an einem Bildschirm arbeitet, je mehr Licht muß ein Monitor abgeben, um der Person ein optimales Arbeiten zu ermöglichen. Eine andere Voraussetzung für ein zügiges Arbeiten an einem Monitor wird nicht im Alter, sondern in frühester Kindheit, in den ersten drei Lebensjahren, geschaffen. Zellen in unserem Gehirn die für das Sehen und Erkennen zuständig sind, werden in dieser Zeit „verdrahtet". Diverse geometrische Grundmuster werden in unseren Gehirnen als Hardwaremuster abgelegt. Alle Zei-

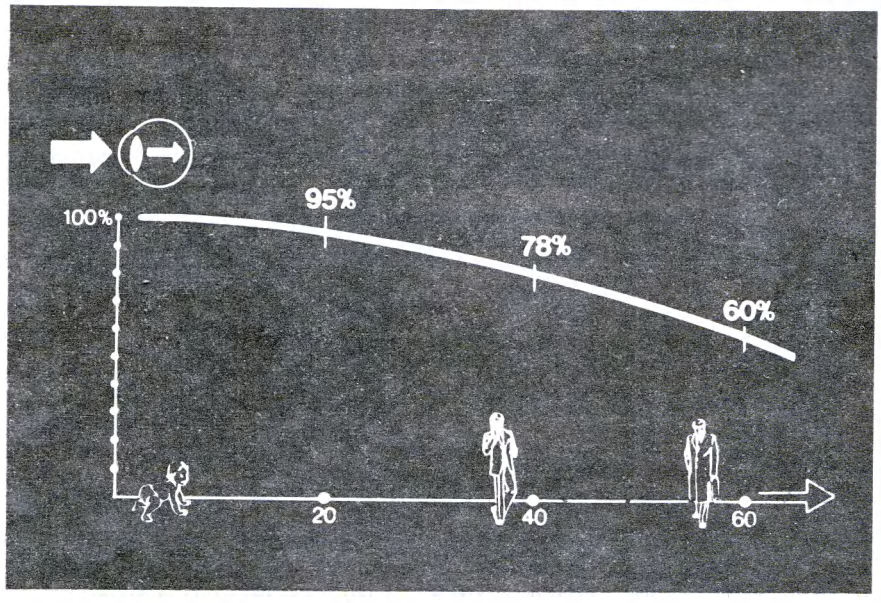


Eine in Phase A auf den Monitor treffende Welle wird bei Durchdringen der 1/4 Lambda-Beschichtung um 90 Grad-Phasen verschoben. Hierauf wird sie in Phase B an der Bildröhrenoberfläche reflektiert. Auf ihrem Weg nochmals um 90 Grad verschoben, ergibt eine Addition dieser Welle (Phase C) mit einer Eintreffenden den Wert 0.





Mit zunehmendem Alter nimmt die Fähigkeit des Nahsehens stark ab.



Die Lichtempfindlichkeit unserer Augen reduziert sich mit steigendem Alter bis auf rund 60 Prozent.

chen und Abbildungen, die nicht in dieser Ablage zu finden sind, muß unser Gehirn mit einer Auswerte-Software aus diesen Grundmustern konstruieren. Dies kostet viel mehr Zeit, als eine Auswertung mit Hilfe einer Hardware.

Die Anzahl und die Typen von Mustern, die unser Gehirn als Hardware ablegt, sind abhängig von der Umgebung, in der wir während der ersten drei Lebensjahre aufwachsen. Umgeben von Hochhäusern und Beton-Architektur bilden gerade Linien und rechte Winkel den Grundstock unserer Wahrnehmungen. Personen, die in freier Natur aufwachsen, übernehmen die dort vorkommende Formenvielfalt und verfügen über einen bedeutend umfangreicheren Vorrat an optischen „Hardwarelösungen".

Wird ein Text vom Bildschirm gelesen, so verhält es sich ähnlich. Von dem Arbeiten mit bedrucktem Papier sind wir es gewöhnt, durchgehende Linien zu erkennen und als Buchstaben zu interpretieren, welche wir dann zu Wörtern zusammensetzen.

Alle Monitore, die eine helle Schrift auf dunklem Grund darstellen, sind nicht in der Lage, kontinuierliche Linien abzubilden.

Deutlich erkennbar werden alle Zeichen aus Punkten zusammengesetzt. Unser Gehirn muß aus diesen Punkten mit Hilfe der „Software" ein Zeichen zusammensetzen. Monitore, die einen schwarzen Buchstaben auf weißem Grund darstellen, bilden kontinuierliche Linien ab. Diese Zeichen können im Gehirn mit Hilfe der „Hardware" ausgewertet werden. Dadurch wird der Lese- und Verständnisvorgang stark beschleunigt.

1974 an der Universität in Toronto/Canada, durchgeführte Testreihen zeigten, daß es bei der damaligen Monitortechnologie 30 Prozent mehr Zeit in Anspruch nahm, einen Text von einem Bildschirm zu lesen, als von einem Blatt Papier. Aktuelle, von Etap initiierte Untersuchungen an der Universität von Groningen zeigen, daß dieses Defizit durch Verbesserungen der Monitortechnologie vollständig ausgeglichen werden konnte. Erstaunlich ist, das ein von einem Monitor gelesener Text sogar schneller und besser verstanden wird, als ein von Papier gelesener.

Gestört wird der Nachbar

Bildschirme, die, bedingt durch eine zu geringe Zeilenfrequenz flimmern, stören nicht so sehr die Person, die direkt davor sitzt. Ein Flimmern ist bedeutend stärker und störender, wenn der Monitor aus spitzem Winkel betrachtet wird. Benachbarte Arbeitsplätze werden hierdurch beeinflußt.

Reflexionen auf einem Monitor sind ein jedem Anwender bekanntes Problem. Durch eine abgestimmte Raumbeleuchtung und eine vernünftige Aufstellung des Gerätes können sie gemindert werden. Die Technik der verwendeten Monitore muß das ihre dazu beitragen, Reflexionen auf ein Minimum zu reduzieren. Eine Möglichkeit bietet ein Ätzen der Bildröhrenoberfläche. Störende Reflexionen verschwinden hierdurch fast vollständig. Gleichzeitig beeinträchtigt das Ätzen aber die Schärfe und somit die Lesbarkeit der Abbildung deutlich.

Eine Beschichtung des Röhreninneren ist derzeit der beste Kompromiß zwischen Reflexion und Schärfe. Ein Viertel-Lambda-Beschichtung heißt eine der neuesten Technologien auf diesem Sektor.

Das bedeutet, daß Fremdlicht auf dem Weg zur Röhrenoberfläche, bedingt durch eine metallische Beschichtung des Röhreninneren, um ein Viertel seiner normalen Wellenlänge von einem Lambda verzögert wird. Reflektierte Strahlen müssen diese Verzögerung zweimal mitmachen und treffen so mit einer Verspätung von einer halben Wellenlänge wieder mit einfallen-

den Lichtstrahlen zusammen. Wird eine Welle, das ist ein Lichtstrahl physikalisch betrachtet, um die Hälfte ihrer Länge verzögert, so spricht man auch von einer Phasenverschiebung von 180 Grad. Das Mischen einer Welle mit einer Welle gleicher Frequenz, aber um 180 Grad gedrehter Phase, führt zu einer Auslöschung derselben. Somit können Reflexionen fast vollständig verhindert werden. Abgesehen davon sind Bildschirme mit weißem Untergrund und schwarzer Schrift sowieso bedeutend weniger anfällig für Reflexionen als solche mit schwarzem Untergrund.

Schlagwort Strahlenbelastung

Strahlenbelastung ist ein in jüngster Zeit oft gebrauchtes Wort. Auch die Bildröhren von Monitoren geben Strahlung ab. Sie entsteht durch die Beschleunigung von Elektronen mit großer Energie und hoher Frequenz. Auf jedem Monitor klebt schließlich ein Schildchen, auf dem die Abschirmung der Röhre dokumentiert ist. Die verbleibende Reststrahlung liegt deutlich unter der Strahlenabgabe anderer Haushalts- und Gebrauchsgegenstände. Die Strahlenbelastung eines Kühlschranks oder Staubsaugers ist, je nach Typ, bis zu zehn mal größer als die eines guten Monitors.

Die bisher gewonnenen Erkenntnisse führen zu einem Pflichtenheft, dem ein Monitor neuester Technologie entsprechen müßte. Resultierend aus Bedingungen, die unser Auge, unsere Arbeitsplatzumgebung und Gesetze der Physik stellen, wurde in den Laboratorien von Etap in Belgien eine Reihe von Ganzseitenmonitoren in den Formaten DIN A4 und A3 entwickelt, die den neuesten Stand der Technik widerspiegeln. Vorgesehen für den Einsatz an IBM-kompatiblen Personal-Computern und bei der MacIntosh-Familie sind sie in der Lage, die Standardmonitore zu ersetzen.

Wir haben die IBM-Version des DIN A4-Monitors Neftis getestet. Bestandteile des Testgerätes waren,nebendem Monitor, eine lange PC-Steckkarte, eine englische Bedienungs- und Installationsanleitung sowie zwei Disketten mit den erforderlichen Bilschirmtreibern.

Bereits nach Überfliegen der ersten Seiten des Handbuchs war klar, daß ein so komplexes Peripheriegerät nur von einem Fachmann installiert werden kann.

Nach zwei Anrufen beim Distributor gelang es uns, die Bildschirmkarte an den Schneider AT 2640 anzupassen. Hierzu müssen zuerst die DIP-Schalter gefunden werden, die die computereigene Bildschirmkarte abschalten. Danach kann der neue Bildschirmtreiber integriert werden. Erscheint beim Einschalten des Computers eine Meldung, die eine erfolgreiche Treiberinstallation quittiert, so steht unter DOS der gesamte, DIN A4 große, Bildschirm zur Verfügung. Ein Directory mit 60 Zeilen ist schon eine imposante Erscheinung.

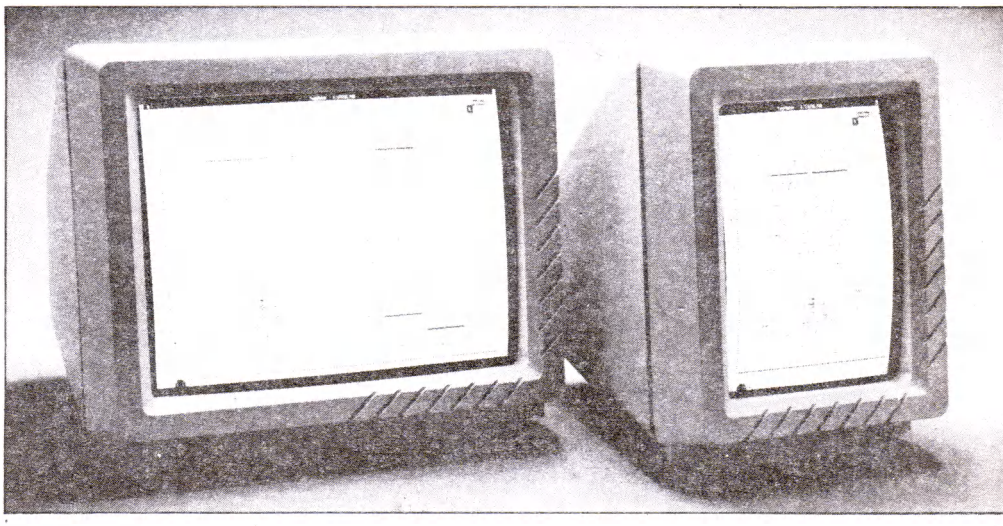
Die nächste Hürde, die es zu überwinden gilt, ist die Anpassung der Software. In einer großen Zahl von Standardanwendungen sind die Etap-Monitore im Installationsmenü bereits enthalten. Weitere Treiber befinden sich auf den mitgelieferten Disketten. Diese Treiber müssen mit einem Editor in die entsprechenden Installationsmenüs integriert werden.

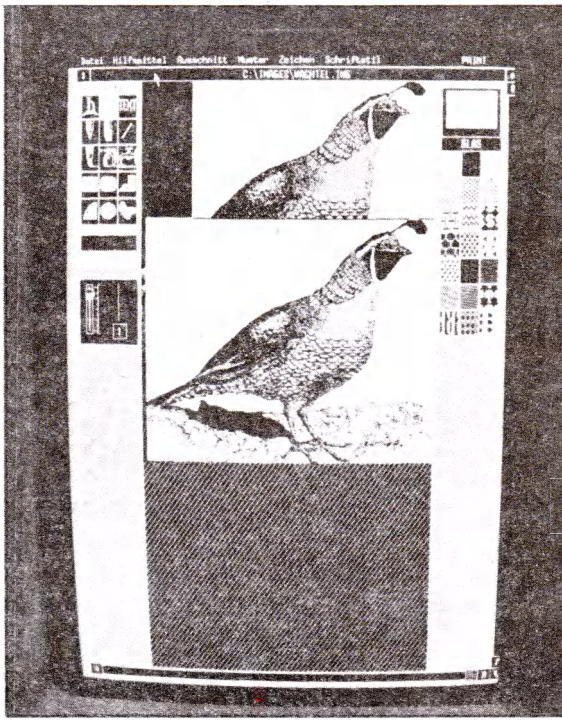
Ausprobiert haben wir den Monitor mit den Benutzeroberflächen GEM und Windows. Auch Word und Wordstar können beispielsweise den kompletten Screen nutzen. Programme, die nicht den gesamten Bildschirm nutzen können, sind trotzdem lauffähig. Die Etap-Monitore emulieren CGA- und Herkulesmodus. So ist es zum Beispiel möglich, zwei Herkulesscreens übereinander und gleichzeitig darzustellen.

Etap-Monitore entsprechen dem neuesten Stand der Technik. Eine sehr gute Abbildungsqualität und Schärfe zeichnen diese Bildschirme aus. Störende Reflexionen sind kaum festzustellen. Eine hohe Bildwiederholfrequenz eliminiert jedes Flimmern. Die Installation dieser Bildschirme sollte auf jeden Fall einem Fachmann überlassen werden. Bei Preisen ab 5000 Mark sollte dieser Service wohl im Preis enthalten sein.

Die Palette lauffähiger, den ganzen Bilschirm nutzender Software ist umfangreich und wird ständig erweitert. Generell gesagt, ist es eine Wohltat, an einem solchen Monitor arbeiten zu können. *LS*

Bezugsquelle:
RFI Elektronik GmbH
Dohrweg 63
4050 Mönchengladbach 1
Tel. 0 21 61/6 00 60





Die Etap-Monitore bilden je nach Modell eine komplette DIN A4- oder DIN A3-Seite ab. Ein Demo-Bild aus GEM macht die Relation zu einem Standardmonitor deutlich.

Atari-Service für Sie bereit

ATARI

7022 L.-Echterdingen

olivetti ATARI ST
Tandon OKI

Matrai computer

Matrai Computer GmbH
Bernhauser Str. 8
7022 L.-Echterdingen
☎ (0711) 797049

8000 München

RIESENAUSWAHL + SUPER SERVICE

SOFTWARE
zu absoluten Niedrigpreisen

Öffnungszeiten unseres Software-Shops
Mo-Fr 10-13 Uhr und 14-18.30 Uhr / Sa 10-14 Uhr

Preis- bzw. Händlerlisten anfordern bei:

philgerma

Barerstr. 32 · D-8000 München 2
Telefonische Bestellannahme und Hotline-Service
089/281228

7030 Böblingen

Partner führender
Micro-Computermarken

Hardware mca Software
COMPUTER CENTER

Norbert Hlawinka
Sindelfinger Allee 1,
7030 Böblingen, Tel. 07031/226015

CH-3415 Hasle-Rüegsau

COMPU-TRADE
Ihr ATARI Spezialist

Emmenstr. 16
CH-3415 Hasle-Rüegsau
0 34 / 61 45 93 auch abends bis 21.00 h

HARD- u. SOFTWARE · BERATUNG · EILVERSAND

2210 Itzehoe

Ihr autorisierter Fachhandel
für Atari, Schneider, Tandon,
Epson, Citizen u.v.m.

EDV-Lehrgänge an AT-kompatiblen Personalcomputern – Schulungsräume für Wochenendseminare mit 16 Arbeitsplätzen.

Der Computerladen
Corlansberg 2 – 2210 Itzehoe
Tel.: 04821/33 90/91

7410 Reutlingen

COMPUTERSHOP
Werner Brock
Federnseestr. 17 - Tel. 07121-34287
Telefax 07121/339779 - D-7410 Reutlingen

COMPUTER STUDIO
Werner Brock
Poststr. 2-4, Tel. 07071/34348
Telefax 0 70 71/3 47 92, 7400 Tübingen

Autorisierter Fachhändler
für Atari, Schneider, Commodore, Panasonic, Kaypro, Sharp, NEC, OKI, STAR ...

CH-4625 Oberbuchsiten

STECTRONIC M. Steck

Electronic-Computer-Shop

Hauptstr. 137
CH-4625 OBERBUCHSITEN
Tel. 062/63 17 27 + 63 10 27

2390 Flensburg

ecl electronic computer laden ohg

Norderstr. 94–96 · D-2390 Flensburg
(04 61) 2 81 81 & 2 81 93

7607 Neuried

CSM
Computer · Software · Marketing
eigener Service · eigene Software

Badstr. 12
7607 Neuried
Tel. 0 78 07/8 22

Filiale:
Hauptstrasse 44
Tel. 0 78 51 / 18 22
7640 KEHL/RHEIN

ELEKTRO-MÜNTZER GmbH

CH-5430 Wettingen

Senn Computer AG

Zentralstrasse 93
CH-5430 Wettingen

Tel. 056 / 27 16 60
Telex 814 193 seco

6900 Heidelberg

HEIDELBERGER
COMPUTER CENTER

micro-data GmbH
Bahnhofstraße 1
6900 Heidelberg
Tel.: 06221/27132

7750 Konstanz

ATARI ★ PC's ★ SCHNEIDER

computer - fachgeschäft
rösler
Rheingutstr. 1 · 0 75 31-2 18 32
D-7750 Konstanz

CH-8021 Zürich

Senn Computer AG

Langstrasse 31
Postfach
CH-8021 Zürich

Tel. 01 / 241 73 73
Telex 814 193 seco

AT 588 32

AT 588 31

AT 588 34

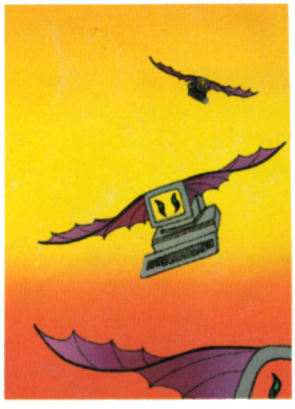
AT 588 33

AT 588 35

AT 588 29

Edition Computer-Kunst

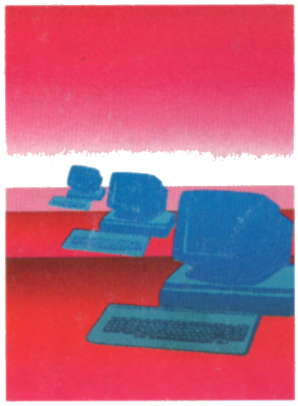
AT 588 30



Computer sind nicht nur Gebrauchsgegenstände. Sie werden zunehmend auch von Künstlern als Motiv entdeckt. Einige besonders gut gelungene Arbeiten haben wir für Sie, unsere Leser, reservieren können. Alle Motive sind strikt auf eine Auflage von 99 Exemplaren limitiert und von der Künstlerin, Sybille Areco, handsigniert und numeriert. Die Exponate werden nach Bestelleingang im 24-Farbendruck von Hand gefertigt, die Vorlage nach dem 99. Druck vernichtet. Unser Angebot: Jedes Motiv nur DM 85,–, zwei Motive DM 150,–, drei DM 210,– und vier Motive nur DM 250,–. Jedes Bild ist 30 x 40 Zentimeter groß und kommt im Passpartout in stabiler Verpackung (im Preis enthalten).
Lieferzeit nach Bestelleingang: ca. drei Wochen.

AT 588 27

AT 588 28



AT 588 25

AT 588 26

Bestellcoupon

Hiermit bestelle ich in Kenntnis ihrer Verkaufsbedingungen folgende Exponate:

Nr.: _____

_____

Ich zahle: (Zutreffendes bitte ankreuzen!)
per beigefügtem Scheck ☐ Schein ☐ Gegen Bankabbuchung am Versandtag ☐

Meine Bank (mit Ortsname) _____

Meine Kontonummer _____

Meine Bankleitzahl _____ **(steht auf jedem Bankauszug)**

Nachname _____ **Vorname** _____

PLZ/Ort _____ **Str./Nr.** _____

Verkaufsbedingungen: Lieferung nur gegen Vorkasse oder Bankabbuchung. Wichtig: Scheckeinreichung und Bankabbuchung erfolgen erst nach dem Versand. Keine Nachnahme möglich. Auf Wunsch Rechnung mit ausgewiesener Mehrwertsteuer.

Unterschrift _____
Bitte ausschneiden und einsenden an

AKTUELL-VERLAG
Heßstraße 90
8000 München 40